Oracle® Fusion Middleware

Administering Oracle Unified Directory

14c (14.1.2.1.0)
G10434-02
April 2025

Oracle Fusion Middleware Administering Oracle Unified Directory, 14c (14.1.2.1.0)

G10434-02

Primary Author: Devanshi Mohan

Contents

10 Understanding the Oracle Unified Directory Schema Model

11 Understanding Root Users and the Privilege Subsystem

12 Understanding the Proxy, Distribution, and Virtualization Functionality

13 Understanding Identity Mapping in Oracle Unified Directory

14 Understanding Data Encryption in Oracle Unified Directory

Part III Basic Administration

15 Starting and Stopping the Server

16 Accessing Oracle Unified Directory Using OUDSM

17 Configuring the Server Instance

18 Managing Directory Data

21 Configuring Load Balancing Using the Proxy

24 Configuring Virtualization

25 Configuring Proxy, Distribution, and Virtualization Deployments

Part V Advanced Administration: Security, Access Control, and Password Policies

26 Configuring Security Between Clients and Servers

27 Configuring Security Between the Proxy and the Data Source

28 Controlling Access To Data

29 Managing Administrative Users

30 Managing Password Policies

33 Managing Directory Schema

36 Tuning Performance

40 Configuring REST API Support

A Appendixes and Glossary

List of Figures

List of Tables

Preface

This guide describes how to manage a deployed Oracle Unified Directory server; including an introduction to basic Oracle Unified Directory concepts and architecture, and step-by-step instructions for performing basic and advanced administrative tasks.

Audience

This guide is intended for administrators of deployed Oracle Unified Directory servers.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Fusion Middleware 14*c* (14.1.2.1.0) documentation set:

- *Release Notes for Oracle Identity Management*
- *Installing Oracle Unified Directory*
- *Configuration Reference for Oracle Unified Directory*
- *Developing Plug-Ins for Oracle Unified Directory*
- *Java API Reference for Oracle Unified Directory*
- *Transitioning to Oracle Unified Directory*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in This Guide?

This preface introduces the new and changed features of Oracle Unified Directory and Oracle Unified Directory Services Manager (OUDSM) since the previous release, and provides pointers to additional information. The information includes the following section:

- New and Changed Features for Oracle Unified Directory 14c (14.1.2.1.0)

New and Changed Features for Oracle Unified Directory 14c (14.1.2.1.0)

Oracle Unified Directory 14*c* (14.1.2.1.0) includes the following new and changed features:

- **JDK Upgrade:** Oracle Unified Directory 14c (14.1.2.1.0) is certified for use with JDK 17, which introduces new features, optimizations, and bug fixes enhancing the overall performance and stability.
- **Berkeley Database Java Edition Upgrade:** Oracle Unified Directory uses Berkeley Database Java Edition for storage and retrieval purposes of OUD. The Berkeley Database Java Edition has been upgraded to version 7.5.18 providing improved functionality and better performance.
- **Deprecating TLS 1.1:** Support for TLS 1.1 and earlier versions has been discontinued. Oracle Unified Directory now requires TLS 1.2 or TLS 1.3 for secure communication.
- **Support for PKCS12 as Default Keystore:** Self-signed certificates are now stored in PKCS12 format by default, providing stronger encryption and improved compatibility. Oracle recommends that you switch the keystore format of the existing security configuration to PKCS12, which is an industry-standard format.

Part I
Overview of Oracle Unified Directory

This part provides an overview of Oracle Unified Directory and the modes in which it can be installed. In addition, it also provides sample deployment scenarios for each server mode, and includes the following topics:

1

Introduction to Oracle Unified Directory

Oracle Unified Directory is an all-in-one directory solution with storage, proxy, synchronization and virtualization capabilities. You will learn about the unique features of Oracle Unified Directory and its architecture in this section.
This section contains the following topics:

- Understanding Oracle Unified Directory
- Overview of Directory Server
- Overview of Proxy Server
- Overview of the Replication Gateway

1.1 Understanding Oracle Unified Directory

Oracle Unified Directory is a comprehensive next generation directory service. It is designed to address large deployments and to provide high performance, and is highly extensive. Oracle Unified Directory is easy to deploy, manage, and monitor.

The following topics provide an overview of Oracle Unified Directory:

- Overview of Oracle Unified Directory Components
- Understanding Oracle Unified Directory Installation Types
- Understanding Oracle Unified Directory Synchronization with Other Directories

1.1.1 Overview of Oracle Unified Directory Components

You can define some components in Oracle Unified Directory for a robust directory server performance.

Oracle Unified Directory components include:

- LDAP directory server, used for storing data

  See Overview of Directory Server.
- Proxy server, where the server acts as an interface between the client and the directory server that contains the data

  See Overview of Proxy Server.
- Replication gateway between Oracle Unified Directory and Oracle Directory Server Enterprise Edition.

  See Overview of the Replication Gateway.

For more information about which Oracle Unified Directory server mode you should use, see Understanding Oracle Unified Directory Installation Types.

1.1.2 Understanding Oracle Unified Directory Installation Types

The mode in which the Oracle Unified Directory server runs depends on how you install the software based on your requirement. You can select the installation type depending on your requirement.

The following installation types are available while installing Oracle Unified Directory:

- About Directory Server Set Up
- About Proxy Server Set Up
- About Replication Gateway Server Set Up

1.1.2.1 About Directory Server Set Up

To create an LDAP directory server that contains directory data, install Oracle Unified Directory as a directory server as described in Setting Up Oracle Unified Directory as a Directory Server in *Oracle® Fusion Middleware Installing Oracle Unified Directory*.

1.1.2.2 About Proxy Server Set Up

If you want the server to act as an interface between the client and the directory server containing the data, then install Oracle Unified Directory as a proxy server. The proxy server does not contain any data. It handles client requests through load balancing or data distribution. See Setting Up Oracle Unified Directory as a Proxy Server in *Oracle® Fusion Middleware Installing Oracle Unified Directory*.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

1.1.2.3 About Replication Gateway Server Set Up

If you want the Oracle Unified Directory server to replicate information between Oracle Unified Directory and Oracle Directory Server Enterprise Edition, then install Oracle Unified Directory as a replication gateway. See Setting Up Oracle Unified Directory as a Replication Gateway in *Oracle® Fusion Middleware Installing Oracle Unified Directory*.

1.1.3 Understanding Oracle Unified Directory Synchronization with Other Directories

You can synchronize Oracle Unified Directory with other directories using Oracle Directory Integration Platform. Oracle Directory Integration Platform consists of a set of services and interfaces that facilitates synchronization and provisioning solutions between the directory and other repositories.

To use Directory Integration Platform to enable synchronization for Oracle Unified Directory, you must enable the Oracle Unified Directory changelog.

Directory Integration Platform synchronization can be described as follows:

- Understanding Synchronization between Oracle Unified Directory and Oracle Internet Directory
- Understanding Synchronization between Oracle Unified Directory and Third-Party Directories

> **Note:**
>
> You can obtain Oracle Directory Integration Platform by installing Oracle Internet Directory release 14.1.2.1.0.

1.1.3.1 Understanding Synchronization between Oracle Unified Directory and Oracle Internet Directory

Oracle Directory Integration Platform 12.2.1.4.0 and higher supports synchronization between Oracle Internet Directory and Oracle Unified Directory. For more information about the synchronization procedure, see Integrating with Oracle Directory Server Enterprise Edition (Connected Directory) in the *Oracle Fusion Middleware Administrator's Guide for Oracle Directory Integration Platform.*

> **Note:**
>
> Oracle Directory Server Enterprise Edition was formerly known as the *Sun Java System Directory Server*. You must replace all references of `SJSDS` in the guide to `OUD` for synchronization to work accurately. You can obtain Oracle Directory Integration Platform by installing Oracle Identity Management release 12.2.1.4.0 or above.

1.1.3.2 Understanding Synchronization between Oracle Unified Directory and Third-Party Directories

To enable synchronization of data between Oracle Unified Directory and third-party directories, you must integrate Oracle Directory Integration Platform with . You can obtain Oracle Directory Integration Platform by installing Oracle Identity Management release 12.2.1.4.0 or above.

1.2 Overview of Directory Server

A directory server provides a central repository for storing and managing information such as identity profiles, user credentials, access privileges, application resource information, and network resource information. The Oracle Unified Directory server is an LDAPv3-compliant directory server written entirely in Java for data storage.

The directory server includes the following high-level functionality:

- *Full LDAPv3 compliance (RFC 4510-4519) with support for numerous standard and experimental extensions*
- *High performance and space effective data storage*
- *Ease of configuration and administration*

  - A highly extensible administrative framework that enables you to customize most of the features listed below.
  - An administration connector that manages all administration traffic to the server. The administration connector enables the separation of user traffic and administration traffic to simplify logging and monitoring, and to ensure that administrative commands take precedence over commands that manipulate user data.
  - A graphical control panel that displays server status information and enables you to perform basic server and data administration.
  - Several command-line utilities to assist with configuration, administration tasks, basic monitoring, and data management. The main configuration utility (`dsconfig`) provides an interactive mode that guides you through most configuration tasks.
- *Advanced replication mechanism*
  - Enhanced multi-master replication across directory server instances
  - Assured replication feature that ensures high availability of data and immediacy of data availability for specific deployment requirements
  - Fractional replication capabilities
  - Support for an external change log that publicizes all changes that occur in a directory server database
- *Extensible security model*
  - Support for various levels of authentication and confidentiality
  - Access to resources based on privileges
  - Advanced access control mechanism
- *Multi-faceted monitoring capabilities*
- *Rich user management functionality*
  - Password policies
  - Identity mapping
  - Account status notification

1.3 Overview of Proxy Server

A proxy server acts as a bridge for requests from clients seeking resources from large-scale networks. Proxy servers enhance performance and security. Oracle Unified Directory support load balancing, failover, data distribution, and global index.

The following topics provide a brief overview of Oracle Unified Directory's proxy component:

- Understanding the Proxy Server
- Understanding the Use of the Proxy Server

1.3.1 Understanding the Proxy Server

The Oracle Unified Directory proxy is an LDAPv3 compliant server that does not store data but routes LDAP requests from clients to the directory servers that are spread across an enterprise.

The proxy is the entry point to a directory service deployment spread over multiple directory servers, multiple data centers, or both. All client requests are routed by the proxy to the

appropriate remote LDAP server. The Oracle Unified Directory proxy component can be used with any LDAP v3-compliant directory server, such as the Oracle Unified Directory server or Oracle Directory Server Enterprise Edition.

To route data requests to the remote LDAP servers, you can configure the proxy component to use either *load balancing* or *data distribution*, or both.

You can deploy the Oracle Unified Directory proxy in very simple configurations, or in more complex, replicated scenarios, using `oud-proxy-setup`. For detailed information about some simple deployments, see Understanding Deployments Using the Proxy Server.

> **Note:**
>
> The proxy component cannot be used directly as a datastore.

As the interface between the client and the remote LDAP server, the proxy provides numerous security features to ensure secure connection if and when required. See Configuring Security Between the Proxy and the Data Source.

For an in-depth presentation of the elements that constitute the Oracle Unified Directory proxy, see Understanding the Proxy, Distribution, and Virtualization Functionality.

1.3.2 Understanding the Use of the Proxy Server

The proxy manages all the connections between a client and a data source (be it a single server, replicated server, or data center). As such, it centralizes all the rules for client connections, including handling load balancing, data distribution and security with the data source.

When you deploy the proxy for load balancing, all requests received by the proxy are routed to one of the remote LDAP servers based on the load balancing algorithm set during deployment. This routing enables you to identify the back-end directory servers that the proxy should communicate with and specify the percentage of total client load each directory server should receive. Once configured, the proxy automatically distributes client queries to different directory servers conforming to the load criteria defined in the configuration.

To deploy a *highly available* directory service, you must have at least two replicated directory servers. To ensure that requests that fail to the first server are treated by the backup server, you must ensure that all the clients know the addresses for both data sources, and are coded to treat a failure on the primary server by re-sending the request to the backup server. The proxy handles the failover and *load balancing* of requests, thereby simplifying high availability and scalability.

Typically, if your deployment used only one server to store all the data, you would have performance issues if your data store was too large. You could resolve this issue by replacing the single server with several servers, and splitting the data across these servers. In this case, each client application would need to know which server to search for its data. With the proxy, there is no need to replicate the distribution information for each application, because the proxy manages the distribution of requests to the appropriate data source. Instead, the client application sends a request to the proxy. The proxy knows which partition holds the requested data and handles the request using *distribution*.

By including the proxy in your deployment, you ease the configuration and management of client applications. The proxy centralizes and handles all requests, ensuring load balancing, distribution of requests, or both.

The proxy also provides a single access point for managing security in a directory service. You can use the proxy to authorize or restrict access to remote directory servers. In addition, to perform maintenance or back up an LDAP server, you can simply modify your proxy deployment to avoid service interruption.

For a description of sample deployments, see Understanding Deployments Using the Proxy Server.

1.4 Overview of the Replication Gateway

A replication server facilitates replication (copying) of data from one Oracle Unified Directory instance to another Oracle Unified Directory server or to another Oracle Directory Server Enterprise Edition (ODSEE) server.

The following topics provide a brief overview of the replication gateway component of Oracle Unified Directory:

- About the Replication Gateway
- Understanding the Role of the Replication Gateway
- Limitations of the Replication Gateway

1.4.1 About the Replication Gateway

Replication is the mechanism that propagates a change made on one directory server to multiple different directories in a replication topology. The replication gateway translates and propagates replication information effectively between directory servers from Oracle Directory Server Enterprise Edition and directory servers from Oracle Unified Directory.

The main purpose of the replication gateway is to facilitate migration from an existing Directory Server Enterprise Edition deployment to an Oracle Unified Directory topology. For this migration to succeed, you must use one of the following versions:

- Any Oracle Directory Server Enterprise Edition since 11*g* Release 1 (11.1.1)
- A Sun Java System Directory Server Enterprise Edition, 6.3.1.1.2 Release (starting with the Oracle Unified Directory 11g Release 2 (11.1.2.3) release)

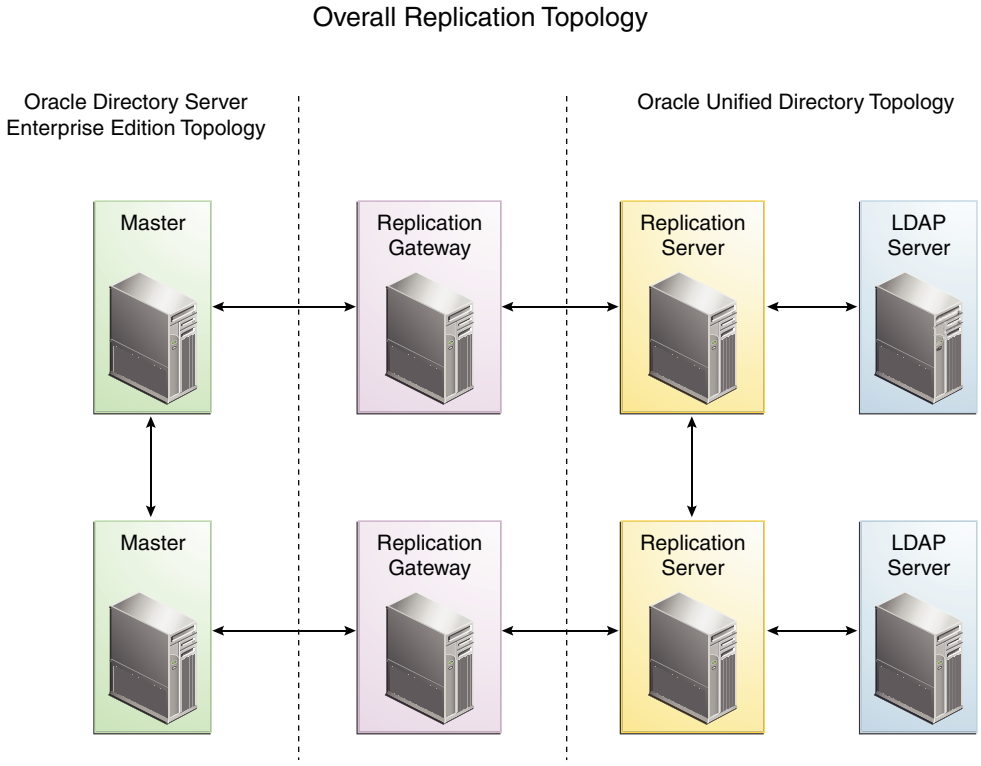
The replication gateway translates the synchronization mechanism specific to each version of the directory, offering two-way replication between the disparate topologies. The replication gateway can be regarded as a *pipe* that propagates updates between heterogeneous replicated topologies. Translations are managed "on the fly" without storing any data on disk.

1.4.2 Understanding the Role of the Replication Gateway

The replication gateway is responsible for propagating changes made on the disparate servers to the entire replication topology. You need replication setup to meet the objectives of high availability and performance.

The following example shows how you can transition an existing Oracle Directory Server Enterprise Edition deployment to an Oracle Unified Directory topology by using the replication gateway between the two topologies.

Figure 1-1 Transitioning an Existing ODSEE Deployment to OUD



Within the overall replication topology, the replication gateway acts as a two-way forwarding server. It propagates modifications from the Oracle Directory Server Enterprise Edition servers to the Oracle Unified Directory replication topology, and from the Oracle Unified Directory servers to the Oracle Directory Server Enterprise Edition replication topology. In each instance, the replication gateway propagates both ways. You can disable changes from being propagated from the Oracle Unified Directory servers to the Oracle Directory Server Enterprise Edition replication topology, according to your transition scenario.

> **Note:**
>
> In a replication architecture, each replication server is connected to every other replication server in the topology.

For high availability, two replication gateway servers are deployed in every transition scenario.

For information about deploying the replication gateway in a migration scenario, see Replicating Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory.

1.4.3 Limitations of the Replication Gateway

Replication is necessary for improving the availability of data across the network. However, there are several limitations to replication that one must be aware of before setting up the replication gateway.

The replication gateway does not manage the following aspects:

- **Data initialization.** Total update is not supported through the replication gateway. To initialize an Oracle Directory Server Enterprise Edition topology with data from an Oracle Unified Directory server, the data must be exported from the Oracle Unified Directory server and then imported to an Oracle Directory Server Enterprise Edition master server.
- **Schema coherency.** The replication gateway does not ensure that schema is coherent across the disparate servers. The administrator must define coherent schema.
- **Feature translation.** The replication gateway does not translate features between the disparate servers, and assumes that the topologies are heterogeneous, regarding features. The best way to handle incompatible features (for example, macro ACIs, CoS, password policies) is to filter out the affected object classes and attribute types before replication occurs.

  The replication gateway does provide a filtering option, for replication *from* Oracle Directory Server Enterprise Edition *to* Oracle Unified Directory. This option enables you to filter out object classes and attribute types that do not apply to Oracle Unified Directory servers. The default values that are configured for filtering account for differences in CoS, roles, password policies, and conflict resolution.
- **Replication Conflict Resolution.** For single-valued attributes, if different values are added simultaneously to the same single-valued attribute, then the Oracle Directory Server Enterprise Edition server and the Oracle Unified Directory server handle the conflict in different ways. The Oracle Directory Server Enterprise Edition server retains the value of the last modify/add operation while the Oracle Unified Directory server retains the oldest value. These values may not always be the same.

2

Understanding Deployment Scenarios Using the Directory Server

In this chapter you will explore some sample configurations for a replicated topology including multiple instances of the Oracle Unified Directory directory server.
It contains the following topics:

- Understanding Small Replicated Topology
- Understanding Multiple Data Center Topology

2.1 Understanding Small Replicated Topology

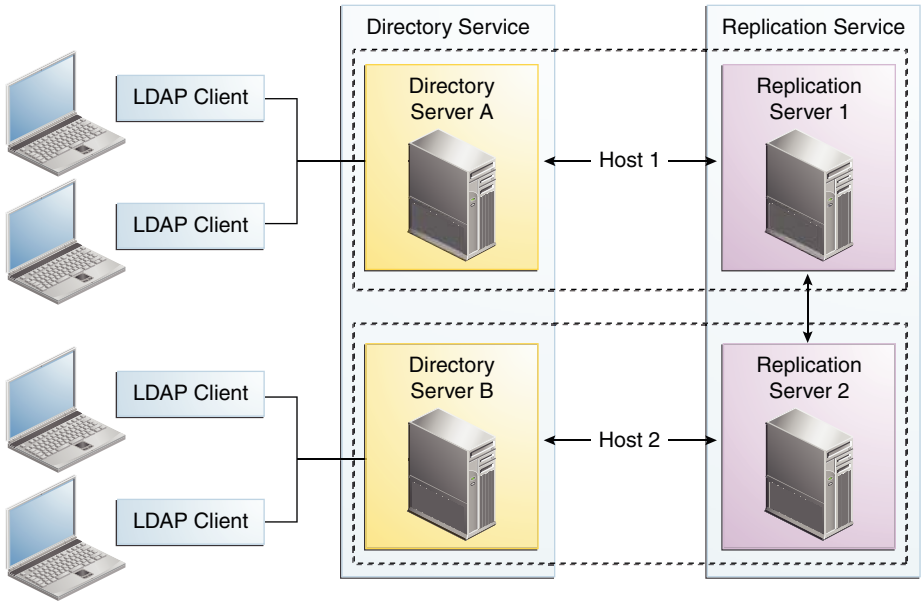
Replicating directory data across servers allows you to reduce the access load on a single server and improve the server response time by providing horizontal read scalability. In addition, you can use replication to ensure availability of data if a system failure occurs.

> **Note:**
>
> You cannot use replication to scale write operations because a write operation to one directory server results in a write operation to every other server in the topology. The only way to scale write operations horizontally is to split the directory data among multiple databases and place those databases on different servers.

The centralized replication model in Oracle Unified Directory separates user data from replication metadata. In this model, the server that stores the user data is called the directory server. The server that stores the replication metadata is called the replication server. This approach simplifies the management of replication topologies and can improve performance.

The following figure shows how you can use replication to ensure availability and to provide read scalability in a small topology.

Figure 2-1 Basic Replication Topology



For small deployments, you can set up replication by putting the replication servers and directory servers on the same system. You can further simplify administration by running the replication server and the directory server on each system in a single process.

This section contains the following topics:

- Role of Directory Servers in a Topology
- Role of Replication Servers in a Topology

2.1.1 Role of Directory Servers in a Topology

Directory servers are primarily responsible for persistence of data, serving client requests, and forwarding changes to specific replication servers. It is essential to understand the different roles to build a robust topology.

When a change is made on a directory server, that server forwards the change to a selected replication server. The replication server then replays the change to other replication servers in the topology, which in turn replays the change to all other directory servers in the topology.

Each directory server contains the following items:

- A list of the suffix DNs to be synchronized
- A list of the replication servers to which each suffix DN can connect

Applications should typically perform reads and writes on the same directory server instance, which prevents those applications from experiencing consistency problems due to loose consistency.

2.1.2 Role of Replication Servers in a Topology

Replication Servers ensure integrity and uniformity of the data by storing replicated data in multiple databases. As a consequence, it reduces load on the network and improves the performance.

Replication servers are responsible for the following tasks:

- Managing connections from directory servers
- Connecting to other replication servers
- Listening for connections from other replication servers
- Receiving changes from directory servers
- Forwarding changes to directory servers and to other replication servers
- Saving changes to stable storage, which includes trimming older operations

Each replication server contains a list of all the other replication servers in the replication topology. Replication servers are also responsible for providing other servers with information about the replication topology. Even the smallest deployment must include two replication server instances, to ensure availability in case one of the replication server instances fails. There is usually no need for additional replication server instances unless the directory service must be able to survive more than one failure at a time, or unless the number of directory server instances must be very large.

> **Note:**
>
> In a replication architecture, each replication server is connected to every other replication server in the topology.

Although replication servers do not store directory data, they are always LDAP servers or JMX servers. Like directory servers, you can configure, monitor, back up, and restore replication servers as described in Understanding the Oracle Unified Directory Replication Model.

2.2 Understanding Multiple Data Center Topology

Replication enables geographic distribution of the directory service by providing identical copies of directory data on multiple servers across more than one data center. The basic principles of a replication deployment outlined in the small topology also apply to multiple data center deployments.

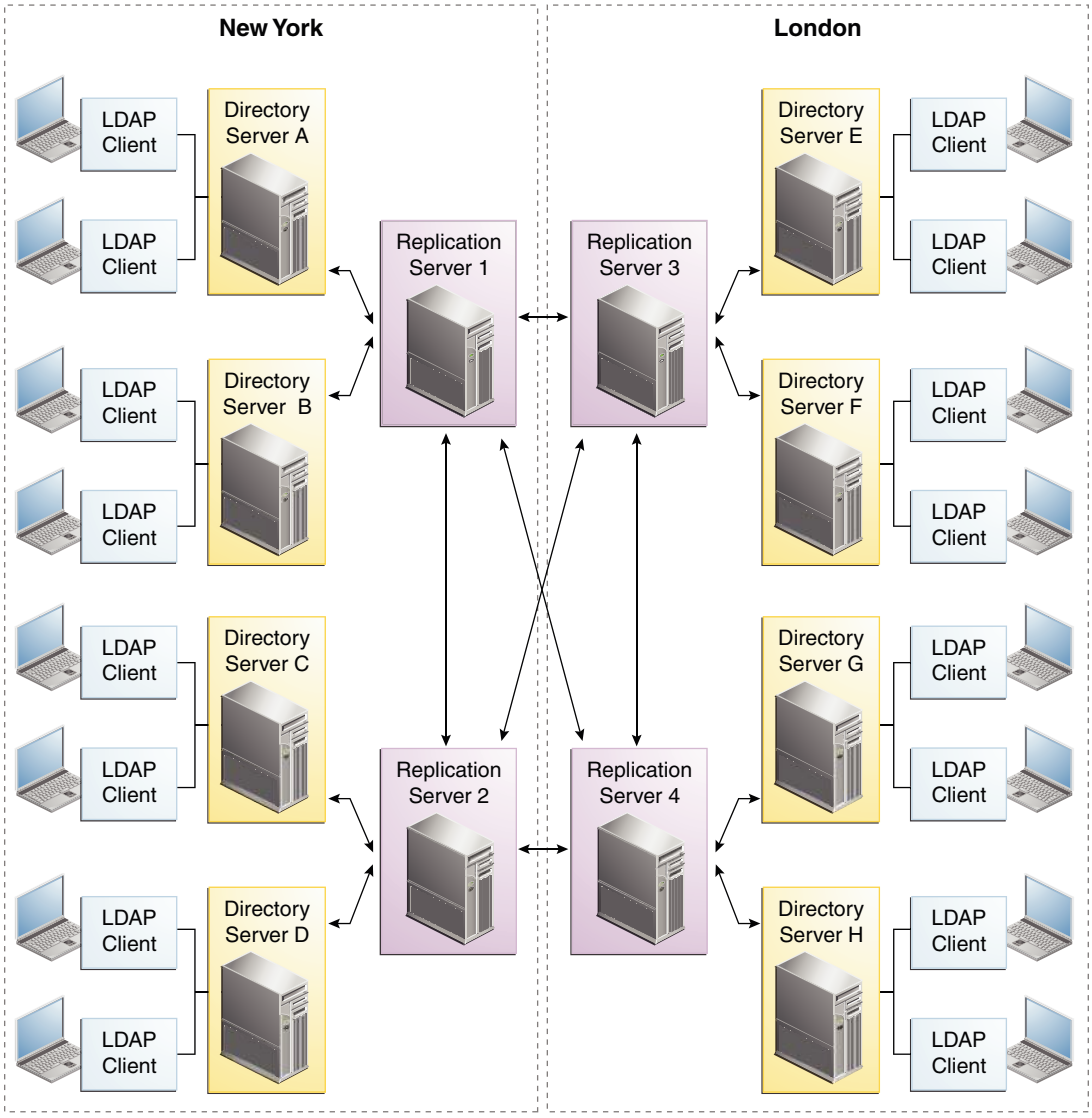
The Oracle Unified Directorydirectory server uses a custom replication protocol that is efficient over a wide area network (WAN). In the following scenario, an enterprise has two major data centers, one in London and the other in New York, separated by a WAN.

This deployment includes two replication server instances for availability in each data center, in case one of the replication server instances fails. The directory servers connect first to local replication servers. Directory servers only access replication servers in another data center if all local replication servers have failed. Client applications always connect to local directory server instances, and perform reads and writes on the same directory server instance.

TheOracle Unified Directory directory server supports an unlimited number of read/write directory servers in a replication topology. The number of directory servers can be scaled according to the read requirements of the organization.

> **Note:**
>
> Increasing the number of directory servers does not scale the number of writes that can be processed because ultimately all servers in the topology must process all the writes. Unless it is acceptable to have a topology that does not converge, the write throughput of the topology is limited to the write throughput of the slowest server.

Figure 2-2 Multiple Data Center Topology



This section contains the following topics:

- Understanding Multiple Data Centers and Replication Groups
- Understanding Multiple Data Centers and the Window Mechanism

2.2.1 Understanding Multiple Data Centers and Replication Groups

Replication groups enable you to organize a replicated topology according to a specific criteria. For instance, the location of a data center. The example in this section demonstrates how to use replication groups across several data centers.

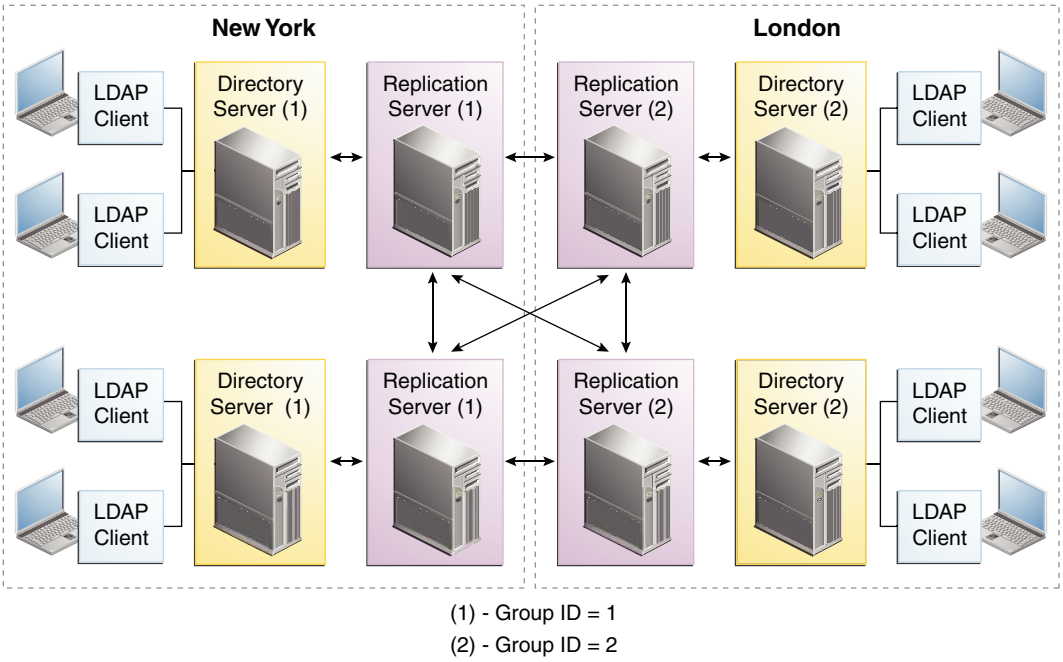
A replication group is identified by a unique ID that is applied to the replication servers and the directory servers in that group. Group IDs determine how a directory server domain connects to an available replication server. From the list of configured replication servers, a directory server first tries to connect to a replication server that has the same group ID as that of the directory server.

This sample deployment illustrates the use of replication groups across multiple data centers. The deployment assumes two data centers, connected by a wide area network (WAN), with the following configuration:

- Each replication server and directory server within a single data center has the same group ID.
- The entire data center has a unique group ID (one group ID per data center).

Figure 2-3 shows a disaster recovery deployment that includes two data centers with different group IDs.

Figure 2-3 Replication Groups Over WAN



In this deployment, each directory server will attempt to connect to a replication server in its own data center, avoiding the latency associated with connection over a WAN. If all the replication servers in a data center fail, the directory server will connect to a remote replication server. This ensures that the replication service is maintained, albeit in a degraded manner (if the connection between data centers is slow). When one or more local replication servers is back online, the directory servers will automatically reconnect to a local replication server.

2.2.2 Understanding Multiple Data Centers and the Window Mechanism

The directory server in Oracle Unified Directory provides a window mechanism, which specifies that a certain number of update requests are sent without one server having to wait for an acknowledgment from the recipient server before continuing.

The window size represents the maximum number of update messages that can be sent without immediate acknowledgment from the recipient server. If the topology spans multiple data centers connected by a network with large latency, it might be worth increasing the window size beyond its default value of 100. To assess whether the window size is the limiting factor in replication throughput, monitor the `current-send-window` and `current-rcv-window` attributes below `cn=monitor`.

If a server publishes a `current-send-window` to another server that is consistently zero or close to zero and the corresponding server publishes a `current-rcv-window` that is higher, it means that all the data are currently in the network. In this case, increasing the window size on the recipient server should increase replication speed and reduce replication delay. These improvements will result in the consumption of more resources on the recipient server.

3

Understanding Deployments Using the Proxy Server

The prime role of a proxy server is to route LDAP requests from clients to directory servers that are deployed in a directory services topology.
The following topics provide some example deployments to help familiarize you with how the proxy server works:

- Understanding Your Proxy Deployment Type
- Supported Proxy Deployments

3.1 Understanding Your Proxy Deployment Type

There are several kinds of scenarios in which you can deploy a proxy server successfully. It is essential to understand the different deployment types to build a network as per your requirement.

The two most common types of deployment with the proxy are:

- Load balancing
- Distribution

To decide which type of deployment to use, consider where and how your data is stored and how much data do you handle.

- If all your data is stored on a replicated data store, then use a deployment with load balancing. See Configuration 1: Simple Load Balancing.
- If your data is partitioned or if you have a large database and want to split your data so that it is partitioned on different data sources, then use a deployment with distribution. See Configuration 2: Simple Distribution.

You can define more complex deployment scenarios that layer load balancing and distribution. The main question is, do you need load balancing, or distribution, or both?

- If you need to deploy data centers in different geographical locations, then you could deploy failover between two load-balanced data centers. See Configuration 3: Failover Between Data Centers.
- If you want to use distribution, but also want the data partitions to be replicated, then you can deploy the proxy server using distribution, which routes to a load balancer. See Configuration 4: Distribution with Load Balancing.
- If you want to use distribution with the data partitions replicated, but for availability and disaster recovery you want the partitions to not only be replicated in one data center but also want to replicate the data centers in two different geographical locations, then you could deploy an architecture similar to Configuration 5: Distribution with Failover Between Data Centers.

> **Note:**
>
> To map entries to a specific partition, add a *global index catalog* to the deployment by using distribution. Adding a global index catalog helps minimize the use of broadcasts. See Configuring Global Indexes Using the Command Line.

3.2 Supported Proxy Deployments

Use the examples in the following sections to familiarize yourself with how the proxy works and the various deployment configurations that are supported by Oracle Unified Directory:

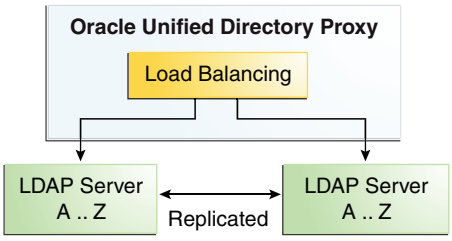
- Configuration 1: Simple Load Balancing
- Configuration 2: Simple Distribution
- Configuration 3: Failover Between Data Centers
- Configuration 4: Distribution with Load Balancing
- Configuration 5: Distribution with Failover Between Data Centers
- Configuration 6: Enterprise User Security
- Configuration 7: Multiple Replicated Proxies
- Configuration 8: Virtualization

3.2.1 Configuration 1: Simple Load Balancing

Load balancing is a strategy to optimize performance by distributing incoming requests across multiple resources. This in turn prevents any single resource from being overloaded. The example illustrates how to set up a simple load balancing.

When you deploy the proxy for load balancing, all requests that the proxy receives are routed to one of the remote LDAP servers. As illustrated in Figure 3-1, the remote LDAP servers are replicated and contain the same data. The number of supported remote LDAP servers is not limited.

Figure 3-1 Simple Load Balancing



The requests are routed to one of the remote LDAP servers based on the load balancing algorithm set during deployment.

The load balancing algorithms are:

- `failover`
- `generic`
- `optimal`
- `proportional`
- `saturation`
- `searchfilter`

For more information about the different load balancing algorithms, see Overview of Load Balancing Using the Proxy.

The algorithm can be bypassed by a client connection affinity. If you set client connection affinity, the proxy uses the load balancing algorithm for the first request, but for the following request will disregard the load balancing algorithm set and will try to reuse the same route for a new operation on the same client connection, for example, depending on the type of client affinity set. See Setting Client Connection Affinity.

The advantages of using load balancing deployment are the high availability of the data, as well as an adapted workload on the remote LDAP servers. For example, if one of the remote LDAP servers in your configuration becomes unavailable, the load balancing will route the request to another remote LDAP server. In this case, the failure is not visible to the client and there is no service disruption.

You can easily configure a simple load balancing deployment during the proxy installation.

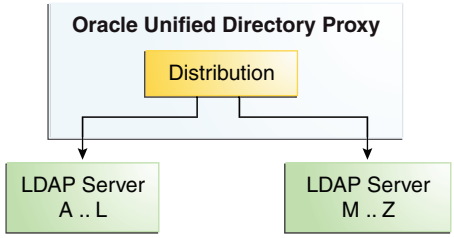
3.2.2 Configuration 2: Simple Distribution

Data distribution allows you to scale your directory across multiple servers. Each server is responsible for only a part of the data. Therefore, a distributed topology can hold large number of data than would be possible with a single server.

When you deploy the proxy for simple distribution, the data is split into partitions. Each partition of data is held on a separate remote LDAP server, as illustrated in Figure 3-2. Here, LDAP Server A...L is a server that holds entries for users whose names start with A through L. Similarly, LDAP Server M…Z holds entries for users whose names start with M through Z. All requests that the proxy receives are routed to the remote LDAP server which contains the appropriate data.

The number of remote LDAP servers onto which the data is partitioned depends on the size of the database that you are splitting. Figure 3-2 shows simple distribution algorithm with two partitions, but you can configure more.

Figure 3-2 Simple Distribution



The requests are routed to one of the remote LDAP servers based on the distribution algorithm set during deployment.

The distribution algorithms are:

- `capacity`
- `numeric`
- `lexico`
- `dnpattern`

For more information about the different distribution algorithms, see Overview of Data Distribution Using the Proxy.

The advantage of a deployment using distribution is that you can scale the number of updates per second. To diminish the number of broadcasts when using distribution, you can add a global index catalog. See Configuring Global Indexes Using the Command Line

A simple distribution deployment can be easily configured during the proxy installation.

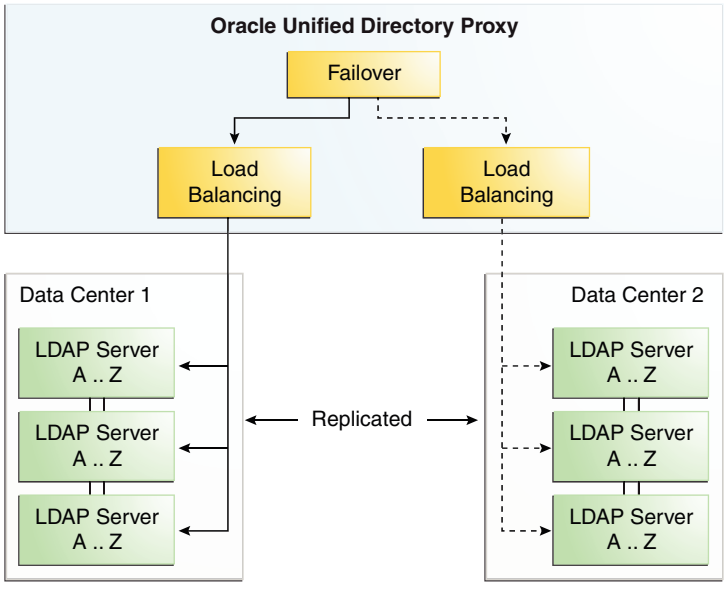
3.2.3 Configuration 3: Failover Between Data Centers

When you configure failover between data centers, you are essentially deploying two levels of load balancers within the proxy. This enables you to enhance the high availability.

Let's examine a failover scenario. In this deployment topology, the data centers are replicated and the remote LDAP servers within the data centers are also replicated. The first load balancing element of the deployment can be either failover or saturation. The example assumes failover algorithm is selected for the initial load balancing element.

As illustrated in Figure 3-3, all of the requests are routed by the failover load balancer through the main route, to a second load balancing element, which sends the request to a server within Data Center 1. Here, LDAP Server A...L is a server that holds entries for users whose names start with A through L. If Data Center 1 goes down or is degraded, then the traffic is routed by the failover load balancer to the backup route, to a server in Data Center 2.

Figure 3-3 Failover Between Data Centers



The requests are routed to the remote LDAP servers within the data centers based on the load balancing algorithm set. The load balancing algorithm can be different for each data center. For example, you can set the load balancing in Data Center 1 as `proportional`, while the load balancing algorithm in Data Center 2 is set as `saturation`.

This type of deployment is typically used when deploying in two geographical areas. This adds high availability of data to a simple load balancing deployment, since not only are the remote LDAP servers replicated, but the data centers are also replicated.

Typically, you would have the two data centers in two different geographical locations. This way, if there was a problem in one location, the data center in the other location would act as backup. Another example would be setting the first load balancer to `saturation`. This way, if Data Center 1 in one geographical location (for example in one time-zone) becomes saturated, the other data center can pick up the excess traffic.

See Also:

- Overview of Load Balancing Using the Proxy for information about the different load balancing algorithms
- Understanding Failover Load Balancing for information about deploying this configuration
- Configuring a Failover Deployment Between Data Centers for an example use case

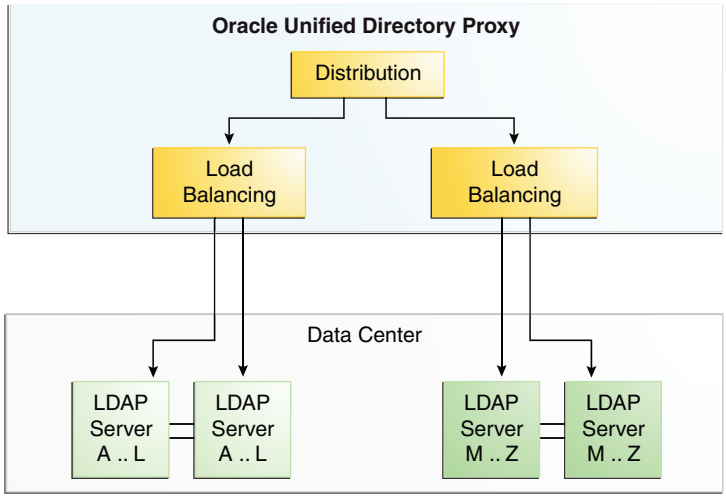
3.2.4 Configuration 4: Distribution with Load Balancing

When you configure distribution with load balancing, data is split into partitions. In addition, the data is replicated on the remote LDAP servers.

In such a situation, the requests sent to the proxy are first distributed to the partition in which the data is stored, then the request is routed to one of the remote LDAP servers, depending on the load balancing algorithm set. The remote LDAP servers holding the partitioned data are replicated.

As illustrated in Figure 3-4, when the proxy receives a requests, it is filtered by the distribution to the correct partition. Here, LDAP Server A...L is a server that holds entries for users whose names start with A through L. Similarly, LDAP Server M…Z holds entries for users whose names start with M through Z. For example, a request for entry with a cn such as `Garry` would be forwarded to partition 1, to the servers with data from `A..L`. The load balancer then forwards the request to one of the replicated remote LDAP servers.

Figure 3-4 Distribution with Load Balancing



The requests are routed to the remote LDAP servers within the data centers based on the load balancing algorithm set. For more information about the different load balancing algorithms, see Overview of Load Balancing Using the Proxy.

The advantages of this deployment are the speed of the updates, because of the distribution of data, and high availability of the data.

See Also:

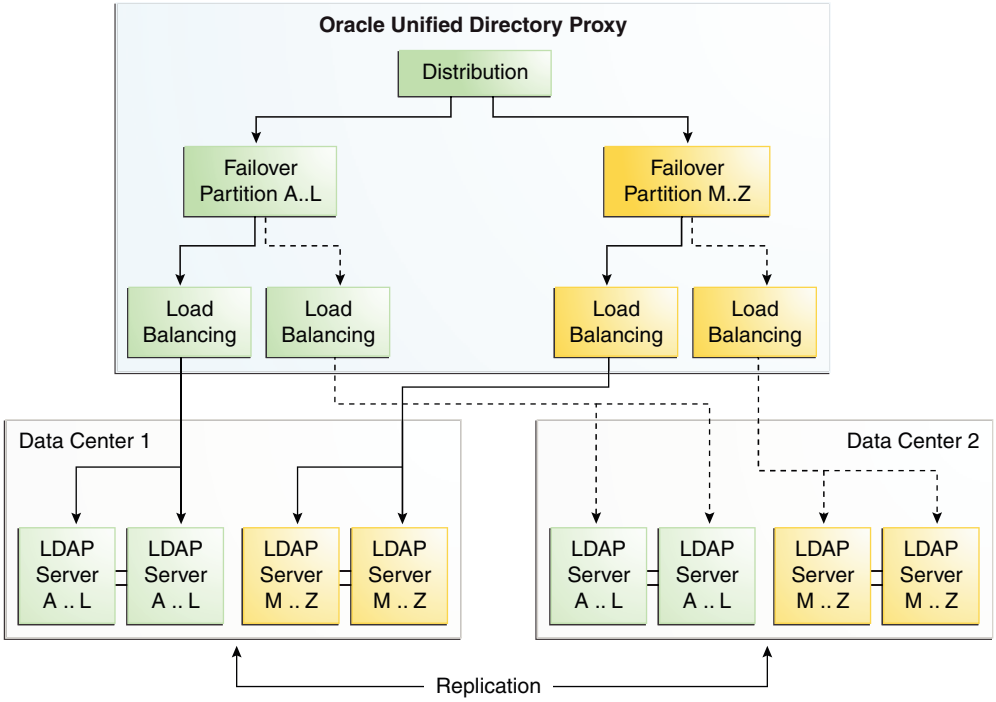
- Overview of Data Distribution Using the Proxy for information about the different distribution algorithms
- Overview of Load Balancing Using the Proxy for information about the different load balancing algorithms
- Understanding the Proxy, Distribution, and Virtualization Functionality for information about deploying this configuration

3.2.5 Configuration 5: Distribution with Failover Between Data Centers

You can configure a topology that includes distribution with failover load balancing. In such a scenario, data is split into partitions between two data centers, where each partition is managed through a failover load balancing route.

Consider the following scenario. As illustrated, in Figure 3-5, not only are the remote LDAP servers holding the partitioned data replicated within the data center, but in addition, the data centers are replicated, with one of the two acting as the backup. Here, LDAP Server A...L is a server that holds entries for users whose names start with A through L. Similarly, LDAP Server M…Z holds entries for users whose names start with M through Z.

Figure 3-5 Distribution with Failover Between Data Centers



In other words, requests sent to the proxy are first distributed to the partition in which the data is stored. For example, a request for entry with a cn such as `Garry` would be forwarded to partition 1. The failover load balancer then forwards the request through the main route, depending on the load balancing algorithm set, to one of the one of the remote LDAP servers holding the data for `A..L`.

In the deployment illustrated in Figure 3-5, Data Center 2 acts as a backup, and is only used on failure of the first data center. However, this same deployment could be configured to use `saturation`, rather than a failover load balancer. This way, if Data Center 1 in one geographical location (for example in one time-zone) becomes saturated, the other data center can pick up the excess traffic.

The advantages of this deployment are the speed of the reads through the distribution algorithm, and the high availability offered since the remote LDAP servers are replicated, and one data center acts as a backup.

See Also:

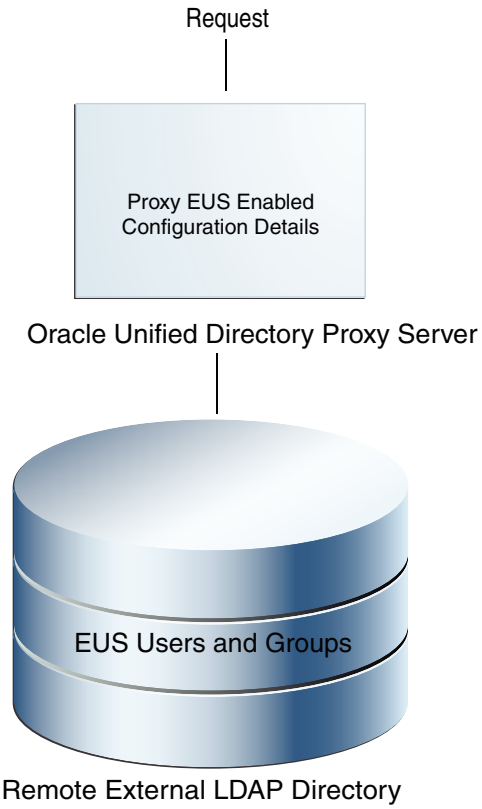
- Overview of Load Balancing Using the Proxy for information about the different load balancing algorithms
- Overview of Data Distribution Using the Proxy for information about the different distribution algorithms
- Configuring a Distribution with Failover Deployment Between Data Centers for an example use case

3.2.6 Configuration 6: Enterprise User Security

When you configure the proxy for Enterprise User Security (EUS), the configuration details are stored locally in the Oracle Unified Directory directory server, and the remote external LDAP directory contains only the Enterprise Users and the Enterprise Groups details.

As illustrated in Figure 3-6 the remote external LDAP directory contains only the Enterprise Users and the Enterprise Groups details.

Figure 3-6 Proxy Enterprise User Security



The requests are routed to one of the remote LDAP servers based on the load balancing algorithm set during deployment.

The load balancing algorithms are:

- `failover`
- `optimal`
- `proportional`
- `saturation`

For more information about the different load balancing algorithms, see Overview of Load Balancing Using the Proxy.

To deploy the proxy for Enterprise User Security, see Integrating Oracle Unified Directory with Oracle Enterprise User Security.

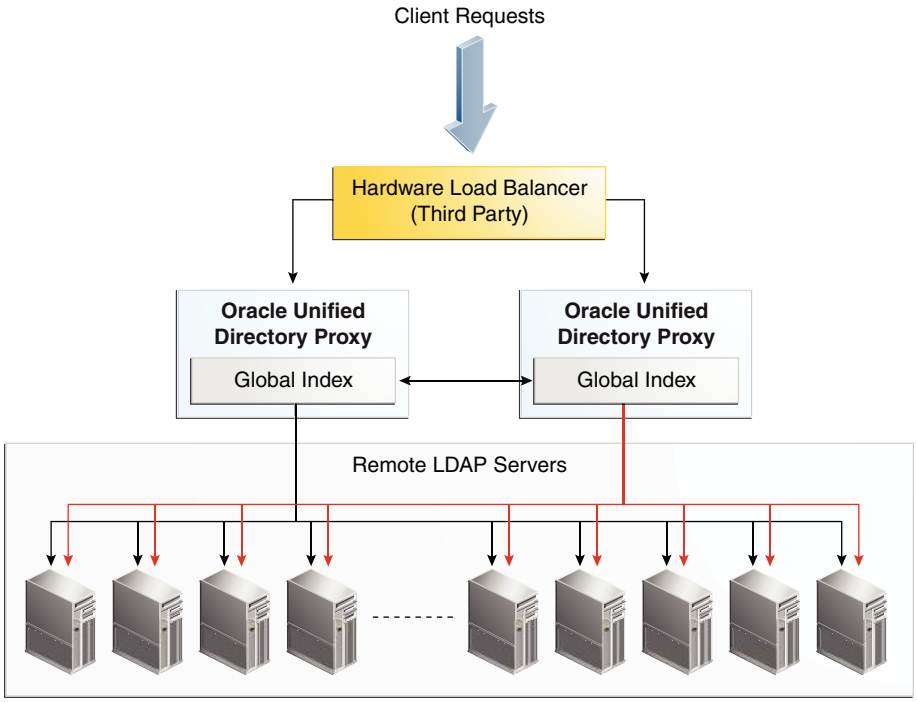
3.2.7 Configuration 7: Multiple Replicated Proxies

You use a hardware load balancer to manage multiple proxy instances on separate physical machines or in different geographical locations.

To prevent a Single Point of Failure, you should ensure that your deployment is redundant. Typically, this can be done by installing a third party hardware load balancer, as illustrated in Figure 3-7.

This example illustrates how to use a hardware load balancer to manage multiple proxy instances.

Figure 3-7 Multiple Proxy Instances



When running multiple proxy instances in a distribution deployment with a global index catalog, the global index catalog should be replicated. For more information about replicating the global index catalog, see Replicating Global Index Catalogs.

To configure this proxy deployment, see Setting Up Oracle Unified Directory as a Proxy Server section in the *Installing Oracle Unified Directory.*

3.2.8 Configuration 8: Virtualization

You can use the virtualization feature of Oracle Unified Directory to create different views of your back-end data and to retrieve data from virtual directories and data sources.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

For example:

- If you have client-side DNs and attributes that do not map with the server-side DNs and attributes, then you can use **DN Renaming** to rename client-side DNs and attributes to match the values in the server.

  For example, your client expects `ou=org, dc=server, dc=com` entries, but the LDAP server contains `ou=people, dc=server, dc=com` entries.

  When the client makes a request, a DN Renaming workflow element applies a DN renaming transformation to the entry DN and to attributes containing either DNs or Name And Optional UIDs syntax. After the result is returned to the client, the DN and attributes are changed back to match what the client requested.
- If you need to rename or replace the relative distinguished name (RDN) values from a source directory to Oracle Unified Directory, then you can use **RDN Changing**.
- If the data structure of your LDAP client application differs from the data structure of an LDAP repository, then you can use transformation to display that physical data in a different way. A transformation performs a specific action in a certain direction (during the request, during the response, or both).

  For example, your client application has a `myuseraccountcontrol` attribute with `activated` and `deactivated` values that you must transform to a `nsAccountLock` attribute with `false` and `true` values on a DSEE (SunONE) back end. You would be required to map the read and write operations.

  You could create a Transformation workflow element that defines where (source or client) Oracle Unified Directory interacts with the data and in which direction the transformation is applied.

> **See Also:**
>
> - Understanding the Proxy, Distribution, and Virtualization Functionality for information about virtualization features for your proxy deployment
> - Configuring Virtualization for information about configuring these features

4
Understanding Mixed Deployments

There are scenarios where it is convenient to deploy the proxy functionality and the Directory Server functionality in a single server instance. This chapter describes the following supported scenarios and the limitations of such deployments:

- Considerations For Mixed Deployment Scenarios
- Example of Pass-Through Authentication Configuration
- Example of Shadow Joiner Configuration

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

4.1 Considerations For Mixed Deployment Scenarios

It is essential that you understand the design considerations and deployment options while designing mixed deployment scenarios.

This section lists those considerations, and contains the following topics:

- Understanding Installation of Oracle Unified Directory as a Directory Server
- Understanding Installation of Oracle Unified Directory as a Proxy

4.1.1 Understanding Installation of Oracle Unified Directory as a Directory Server

This section is intended to help you understand the considerations to bear in mind when you install Oracle Unified Directory as a directory server.

You must keep the following points in mind while you install Oracle Unified Directory as a directory server using the `oud-setup` command:

- You can only use Local Backends, Kerberos, EUS, and Pass-Through Authentication workflow elements.
- Virtual ACIs are not supported.
- You can use all the advanced features of the Local Backends, such as password policy, group, collective attributes, virtual attributes, privileges, referential integrity, password storage, seven bit, and so on.
- You can use replication for Local Backends workflow element.

4.1.2 Understanding Installation of Oracle Unified Directory as a Proxy

If you install Oracle Unified Directory as a proxy server, then you can achieve pass-through authentication or Join features using the workflow elements associated with them. However, you need to understand the considerations while doing so.

You must keep the following points in mind:

- You can use all the non-local workflow elements, such as LDAP Proxy, Join, Renaming, Transformation, RDN changing, AD paging, Distribution, and Load Balancing.
- You can either use pass-through authentication or EUS.
- You can use Local Backends as Join Participant.
  - The advanced features of the Local Backends is not supported.
  - You can use replication for a Local Backends workflow element.
  - ACIs defined for Local Backends workflow element are not compatibles with DN mapping at Join or pass-through authentication level, therefore you must use virtual ACIs.
- You can use virtual ACIs, but bind rules can only use bind DN. For more information about bind rules, see About the Virtual Access Control Instructions Syntax.
- You can replicate Virtual ACIs back end.

4.2 Example of Pass-Through Authentication Configuration

Pass-through authentication is a strategy in which a directory server consults another to authenticate bind requests. This enables you to administer user and configuration information on separate instances of Directory Server.

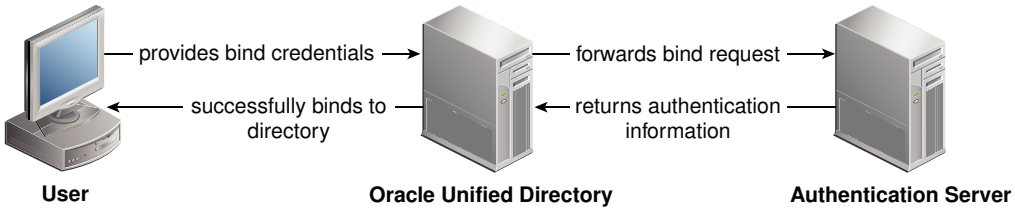
You use pass-through authentication mechanism when the client attempts to bind to the directory server and the user credentials for authenticating are not stored locally, but instead in another remote directory server known as the authentication (Auth) server. This in turn implies, that when the user tries to authenticate, the BIND request is forwarded to the remote LDAP server, but other operations are handled locally by directory server. Such a deployment is called pass-through authentication.

Figure 4-1 depicts the pass-through authentication mechanism.

The user password is stored in a remote LDAP server, but all the other attributes of the user entry are stored in locally in Oracle Unified Directory.

> **See Also:**
>
> - Understanding Pass-Through Authentication for information about configuring pass-through authentication
> - Optimizing Search Results From a Virtual Directory for information about configuring pass-through authentication using a Join workflow element

Figure 4-1 Pass-Through Authentication Mechanism



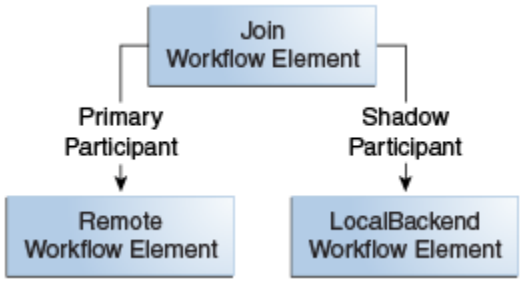
4.3 Example of Shadow Joiner Configuration

The Shadow Joiner allows you to store entries in a source such as an LDAP or Database stores that requires a schema extension for the remote data store, but the schema extension is not possible either for business or technical reasons.

The Shadow joiner allows you to store the extended attributes in another store, known as the shadow directory, such as the Local Backend workflow element. See Shadow Joiner Type.

Figure 4-2 illustrates a Shadow join configuration. The remote workflow element contains the user entry, whereas the Local Backend contains locality and description attribute and Join with remote server.

Figure 4-2 Shadow Joiner Configuration



Part II

Oracle Unified Directory Concepts and Architecture

This part describes the details of how Oracle Unified Directory works. These chapters describe the architecture of Oracle Unified Directory and the various components that comprise that architecture.

In general, you do not need a thorough understanding of all of these concepts to administer Oracle Unified Directory, but an overview of these chapters might help to make your administration easier.

This part includes the following chapters:

- Understanding Oracle Unified Directory Concepts and Architecture
- Understanding Oracle Unified Directory High Availability Deployments
- Understanding the Oracle Unified Directory Replication Model
- Understanding the Oracle Unified Directory Indexing Model
- Understanding Access Control Model in Oracle Unified Directory
- Understanding the Oracle Unified Directory Schema Model
- Understanding Root Users and the Privilege Subsystem
- Understanding the Proxy, Distribution, and Virtualization Functionality
- Understanding Identity Mapping in Oracle Unified Directory
- Understanding Data Encryption in Oracle Unified Directory

5

Understanding Oracle Unified Directory Concepts and Architecture

Oracle Unified Directory is a next-generation unified directory solution that integrates storage, synchronization, and proxy functionality to help you manage the critical identity information that drives your business applications. These capabilities enable you to meet the evolving needs of an enterprise architecture.
The following topics provide conceptual descriptions of the basic components of Oracle Unified Directoryand discusses Oracle Unified Directory architecture:

- Understanding Oracle Unified Directory Components
- Overview of Oracle Unified Directory Architecture

5.1 Understanding Oracle Unified Directory Components

Oracle Unified Directory integrates three key components: Network Groups, Workflows, and Workflow Elements. It is imperative to understand the role of each component to gain insight into the complete functionality of the product.

This section provides an overview of each component and contains the following topics:

- Understanding Network Groups
- Understanding Workflows
- Understanding Workflow Elements

5.1.1 Understanding Network Groups

Network groups are the entry point of all client requests handled by Oracle Unified Directory.

Network groups are described in the following topics:

- About Network Groups
- Using Network Group Criteria to Route to Different Workflows
- Using Network Group QOS Policy to Filter Requests

5.1.1.1 About Network Groups

Network groups handle all client interactions and dispatch them to local back end workflow or proxy workflow based on some norms. You need to understand those standards to define a robust configuration.

The network groups makes use of the following standards to handle client interactions:

- Criteria

  Criteria can include security authentication level, port number, client IP mask, client bind DN, bind ID, domain name, and other criteria.

- Quality of Service (QoS) policies

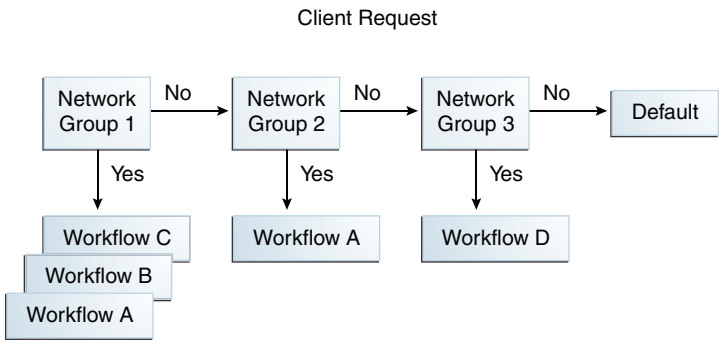
QoS policies can include LDAP referral policy, request filtering, client connection affinity, and resource limits.

You can define more than one network group, each with different properties and different priorities. However, an incoming client connection can only be attached to one network group at a time. An incoming client connection is attached to the first network group for which the connection complies with the criteria defined for that network group.

The client connection is assessed by each network group, in order of priority, until it complies with all the criteria of that network group. As illustrated in Figure 5-1, the request is first sent to the network group with the highest priority: `Network Group 1`. `Network Group 1` assesses if the request matches all the required criteria. If it does not match all of the criteria, it forwards the request to the next network group in the list: `Network Group 2`.

If the request matches all the properties of a network group, the network group assesses if the client connection matches the QoS policies of that network group. If it matches the QoS policies, it is routed to the associated *workflow*.

Figure 5-1 Network Group Selection



A network group can be associated with one or more workflows, each workflow corresponding to a different naming context. For more information about workflows, see Understanding Workflows. If the client connection matches the criteria of a network group, but not the QoS policies of that network group, the connection is not forwarded to the workflow, nor is it sent to the next network group. Instead, an error is returned, indicating the QoS policy that caused the error.

If a network group has no workflows attached to it, the request is not handled. Instead, the server returns an error message of the sort: `No such entry`.

For information about managing network groups, see Configuring Network Groups Using `dsconfig`.

5.1.1.2 Using Network Group Criteria to Route to Different Workflows

To use the network group criteria to route different workflows, perform the steps described in this section.

Assume an Oracle Unified Directory configuration with the following network groups:

1. Configure network group 1 as follows:

   `Network Group 1`: criteria set with bind DN `**,dc=example,dc=com`

This network group is associated with `Workflow 1`, with naming context `dc=example,dc=com`

2. Configure network group 2 as follows:

   `Network Group 2`: criteria set with bind DN `**,dc=test,dc=com`

   This network group is associated with `Workflow 2`, with naming context `dc=test,dc=com`

Depending on the bind DN, a search would be routed through `Network Group 1` or `Network Group 2`. For example, if the bind DN was `uid=user.1,dc=test,dc=com`, then request would not be accepted by `Network Group 1`, but would be forwarded to and accepted by `Network Group 2`, and forwarded to `Workflow 2`.

5.1.1.3 Using Network Group QOS Policy to Filter Requests

To use the network group QOS policy set to filter requests, perform the steps described in this section.

Assume an Oracle Unified Directory configuration with the following network groups:

1. Configure network group 1 as follows:

   `Network Group 1`: criteria set with bind DN `**,ou=admin,dc=example,dc=com`

   QoS policy set with resource limits size limit=0, time limit=0. Therefore, for admin group, there are no limits.

   This network group is associated to `Workflow 1`, with naming context `dc=example,dc=com`.

2. Configure network group 2 as follows:

   `Network Group 2`: criteria set with bind DN `**,dc=example,dc=com`

   QoS policy set with resource limits `size limit=100`, `time limit=30 s`. Therefore, for all connections other than admin group, there are limits set on the resources used.

   This network group is also associated to `Workflow 1`, with naming context `dc=example,dc=com`.

Therefore, if the bind DN is `dc=example,dc=com`, then the requests will be forwarded to `Workflow 1`. The QoS policy set for `Network Group 2` gives restricted access to `Workflow 1`, for anyone that is not `admin`. Anyone who binds as `admin` will access `Workflow 1` through `Network Group 1`, and will have no limitations on resource limits.

5.1.2 Understanding Workflows

A workflow represents the flow of data. It comprises workflow elements and their associated connections.

A workflow is defined by a *naming context* (base DN) and a workflow element that defines how Oracle Unified Directory should handle an incoming request.

A workflow must be registered with at least one network group, but can be attached to several network groups.

To learn more about workflow, you must review the following topics:

- About Workflows
- Using Network Groups to Route to Different Workflows

5.1.2.1 About Workflows

A workflow is the link between the network group and the naming context (suffixes). It defines the naming context that will be accessible for a given network group, when handling a request to a load balancing or distribution configuration.

A network group can point to *several* workflows if the naming contexts of the workflows are different. However, several network groups can point to the *same* workflow when the network group QoS policies are different, but the naming context of the workflow is the same.

Each workflow is associated with an access control group, which defines the list of ACIs that apply to operations handled by this workflow. By default, an access control group known as `Local Backends`, exists. This access control group contains all ACIs coming from user data. You cannot delete it. You can also add virtual ACIs in this group, which implies that you must specify `Local Backends` as the access control group for the workflow for which virtual ACIs are disabled. You can specify any access control group for the workflow where virtual ACIs are enabled. For more information about ACIs, see Understanding Access Control Model in Oracle Unified Directory.

5.1.2.2 Using Network Groups to Route to Different Workflows

Use network groups to route to several different workflows.

Assume an Oracle Unified Directory configuration with the following network groups (as illustrated in Figure 5-1), where:

- `Network Group 1` with a bind DN of `**,l=fr,dc=example,dc=com`

  This network group is associated to `Workflow 1`, with naming context `l=fr,dc=example,dc=com`

- `Network Group 2` with a bind DN of `**,l=uk,dc=example,dc=com`

  This network group is associated to `Workflow 2`, with naming context `l=uk,dc=example,dc=com`

- `Network Group 3` with a bind DN of `**,dc=example,dc=com`

  This network group is associated to `Workflow 1` and `Workflow 2`, with naming context `dc=example,dc=com`

A search with bind DN `**,l=uk,dc=example,dc=com` would be handled by `Network Group 2` and sent to `Workflow 2`.

A search with bind DN `**,dc=example,dc=com` would be handled by `Network Group 3` and sent to `Workflow 1` and `Workflow 2`.

5.1.3 Understanding Workflow Elements

Workflow elements are part of a routing structure. Each workflow contains at least one workflow element.

Oracle Unified Directory supports several different types of workflow elements:

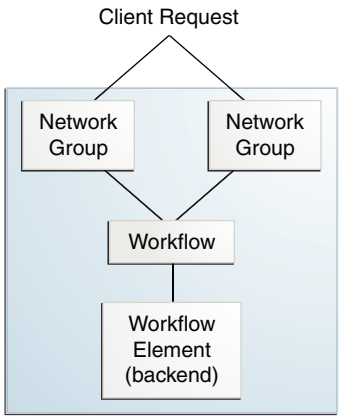
- Leaf workflow elements: This type comprises the Local Backend workflow elements and proxy workflow elements.
- Routing workflow elements: This type comprises the load balancing workflow elements and distribution workflow elements.

- Virtual workflow element: This type comprises the DN renaming workflow elements, RDN changing workflow elements, and Transformation workflow elements.
- EUS workflow element: This type comprises the Enterprise User Security (EUS) workflow elements.
- EUS context workflow element: This type comprises the EUS context workflow elements.
- LDIF workflow element: This type comprises the LDIF Local Backend workflow elements.
- Memory backend workflow element: This type comprises the memory local backend workflow elements.

For a directory server, the workflow element is the DB Local Backend, as illustrated in Figure 5-2.

For a proxy server, the workflow elements can be chained with load balancing workflow elements or distribution workflow elements that act as a pointer, routing the request along a specific path. The proxy workflow element provides direct access to the remote data source.

Figure 5-2 Client Request for a Directory Server



Oracle Unified Directory has several preconfigured workflow elements that should not be modified or deleted.

5.2 Overview of Oracle Unified Directory Architecture

Oracle Unified Directory is a Java-based directory service that provides storage, synchronization, proxy, and virtualization features. The unified solution provides architecture flexibility and optimization, enhances application deployments, and reduces total cost of ownership.

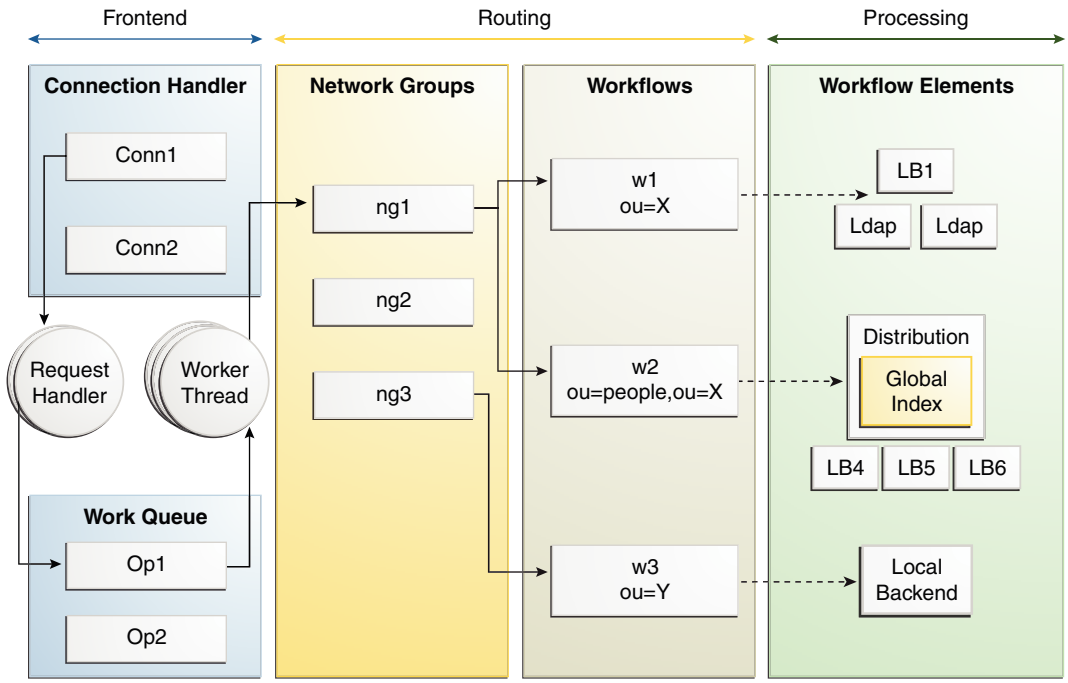
The example in this section illustrates how the components of Oracle Unified Directory work together to provide a comprehensive solution to the industry.

As illustrated in Figure 5-3, a client request is managed by Oracle Unified Directory before being forwarded to the data source. In this scenario, there are three network groups, such as ng1, ng2, and ng3. The first network group ng1 contains two workflows while ng3 contains a single workflow. A workflow is defined by a suffix. The suffix for `w1` is `ou=X` and a workflow points to a tree of workflow elements. The tree of workflow elements determines the processing to apply on an operation.

A client request pursues the following path:

1. The request handlers place the incoming LDAP requests in the work queue from where the worker thread grabs them.
2. The operation is routed to a network group based on the network group criteria assigned. An operation must comply with the network group QoS policies regardless of the server profile, directory server or proxy server.
3. The network group forwards the operation to a workflow, which defines the naming context. The determination of the workflow is based on the match between the request base DN and the workflow naming context.
4. The workflow forwards the operation to its tree of workflow elements, which defines how to treat the request. The content of the tree of workflow elements depends on the server profile as follows:
   - For a directory server, you can only configure the workflow element as the local backend workflow element (a storage).
   - For a proxy server, you can configure the workflow element as a distribution workflow element, a load balancing workflow element, a DN renaming workflow element, or an LDAP proxy workflow element.
5. After the request has gone through the assigned processing, the request is sent to the data source.

Figure 5-3 High-Level Presentation of Oracle Unified Directory Components



6

Understanding Oracle Unified Directory High Availability Deployments

It is essential to have a reliable service as most enterprise applications depend on a directory server. You can deploy Oracle Unified Directory to ensure high availability between directory server instances or between groups of directory server instances.
The following topics explain high availability and how Oracle Unified Directory features help provide continued service if a system failure occurs:

6.1 Overview of High Availability

As more and more businesses and mission-critical applications connect with identities being centrally managed, it has become imperative to have LDAP service available all the time. High availability with performance has become the distinguishing feature of all extranet and enterprise deployments.

High availability is a system design approach and its associated implementation that ensures an agreed level of operational performance during a given measurement period for your directory service.

Agreed service levels vary from one organization to another. Service levels also depend on several factors such as the time of day systems are accessed, whether systems can be brought down for maintenance, and the cost of downtime to the organization. Failure or downtime in this context, is defined as periods when a system is unavailable and prevents from providing the agreed level of service.

Oracle Unified Directory provides elaborate cost-effective and easy-to-use high availability features, which eliminate the downtime and maximize the time when the system is available.

6.2 Understanding Availability and Single Points of Failure

Oracle Unified Directory deployments that provide highly available service can recover from failures and maintain service within agreed level of service. With a high availability deployment, component failures might impact individual directory queries but does not result into a complete system failure.

A single point of failure (SPOF) is a system component, which on failure renders an entire system unavailable or unreliable. When you design a highly available deployment, you identify potential SPOFs and investigate how to mitigate these SPOFs.

The following topics discuss availability and single points of failure:

- Understanding the Approach to Mitigate SPOFs

6.2.1 Understanding Types of SPOFs

A single point of failure (SPOF) is a system component, which on failure renders an entire system unavailable or unreliable. When you design a highly available deployment, you identify potential SPOFs and investigate how to mitigate these SPOFs.

You can divide SPOFs into the following categories:

- About Hardware Failure
- About Software Failure

6.2.1.1 About Hardware Failure

You can broadly categorize the hardware SPOFs as follows:

- Network failures
- Failure of the physical servers on which Directory Server or Directory Proxy Server are running
- Hardware load balancer failures
- Storage subsystem failures
- Power supply failures

6.2.1.2 About Software Failure

You can categorize Directory server or proxy server failures as follows:

- Slow response time
- Write overload
  - Maximized file descriptors
  - Maximized file system
  - Poor storage configuration
  - Too many indexes
- Read overload
- Cache issues
- CPU constraints
- Replication issues
  - Out of sync
  - Replication propagation delay
  - Replication flow
  - Replication overload
- Large wildcard searches

6.2.2 Understanding the Approach to Mitigate SPOFs

A SPOF is hardware or software component that could cause the entire system to become non viable and unusable if the component fails. Redundancy is a strategic approach to handle SPOF.

You can implement redundancy to ensure that failure of a single component does not cause an entire directory service to fail. Redundancy involves providing redundant software components, hardware components, or both. Examples of this strategy include deploying multiple, replicated instances of Directory Server on separate hosts and using redundant arrays of independent disks (RAID) for storage of Directory Server data. Redundancy with replicated Directory Servers is the most efficient way to achieve high availability.

6.3 Overview of Redundancy for High Availability

To ensure reliability and continued services for directory service, you must maintain a high level of system availability, with a seamless transition to redundant systems during a system failure.

Redundancy works for both Directory and proxy servers and allows you to mitigate:

- Hardware failures, because the traffic can be redirected to another hardware component.
- Software failures, when the failure cannot be reproduced systematically.

Redundancy handles failure in the following ways:

- Understanding Redundancy at the Hardware Level
- Understanding Redundancy at Directory Server Level Using Replication
- Understanding the Use of Directory Proxy Server as Part of a Redundant Solution
- Understanding the Use of Application Isolation for High Availability
- Understanding How to Use the Replication Gateway for High Availability

6.3.1 Understanding Redundancy at the Hardware Level

Hardware happens to be the most crucial SPOF. Hardware could be any part in the network that handles network traffic, controls the system, or manages authentication.

This section provides an overview of hardware redundancy.

> **Note:**
>
> Providing comprehensive information on this topic is beyond the scope of this book. However, there are many publications available that concern using hardware redundancy for high availability, such as "Blueprints for High Availability" published by John Wiley & Sons, Inc.

Failure at the network level can be mitigated by having redundant network components. When designing your deployment, consider having redundant components for the following:

- Internet connection

- Network interface card
- Network cabling
- Network switches
- Gateways and routers

You can mitigate the hardware load balancer as an SPOF by including a redundant hardware load balancer in your architecture.

You can mitigate against SPOFs in the storage subsystem by using redundant server controllers. You can also use redundant cabling between controllers and storage subsystems, redundant storage subsystem controllers, or redundant arrays of independent disks.

If you have only one power supply, loss of this supply could make your entire service unavailable. To prevent this situation, consider providing redundant power supplies for hardware, where possible, and diversifying power sources. Additional methods of mitigating SPOFs in the power supply include using surge protectors, multiple power providers, and local battery backups, and emergency local power generators.

Failure of an entire data center can occur if, for example, a natural disaster strikes a particular geographic region. In this instance, a well-designed multiple data center replication topology can prevent a distributed directory service from becoming unavailable. See Sample Topologies Using Redundancy for High Availability.

6.3.2 Understanding Redundancy at Directory Server Level Using Replication

Replication is a common method used to implement redundancy in Oracle Unified Directory Servers. Replication provides a failover system to implement redundancy.

Redundant solutions are usually less expensive, easier to implement, and easier to manage than clustering solutions. In a clustering model, you often have to configure at least two servers to serve the same application workload, where one node is active while the other is passive, on standby.

Be aware that replication, as part of a redundant solution, has numerous functions other than availability. While the main advantage of replication is the ability to split the read across multiple servers, you must balance this advantage with the task to manage the additional servers. Replication also offers scalability on read operations and, with proper design, scalability on write operations, within certain limits. See Understanding the Oracle Unified Directory Replication Model.

The SPOFs described in About Software Failure can be mitigated by having redundant instances of Directory Server. This involves the use of replication. Replication ensures that the redundant servers remain synchronized, and that requests can be rerouted with no downtime.

Replication is used to prevent the loss of a single server from causing your directory service to become unavailable. A reliable replication topology ensures that the most recent data is available to clients across data centers, even when a server fails. At a minimum, your local directory tree must be replicated to at least one backup server. Directory architects recommend you to replicate three times per physical location for maximum data reliability. When the data is replicated at least thrice then, if a directory server failure occurs, the configuration remains highly available and protected. In deciding how much to use replication for fault tolerance, consider the quality of the hardware and networks used by your directory. Unreliable hardware requires more backup servers.

The Oracle Unified Directory replication model is a loosely consistent, multi-master model. In other words, all directory servers in a replicated topology can process both read and write operations. See Understanding the Oracle Unified Directory Replication Model.

Do not use replication as a replacement for a regular data backup policy. Replication is designed to maintain service within a given service level agreement. It is not designed to protect against incorrect data stored in the directory by applications or users. See Backing Up, Purging, and Restoring Data.

To maintain the ability to read data in the directory with the expected Service Level Agreement, a suitable load balancing strategy must be put in place. Both software and hardware load balancing solutions exist to distribute read load across multiple replicas. Each of these solutions can also determine the state of each replica and to manage its participation in the load balancing topology. The solutions might vary in terms of completeness and accuracy.

To maintain write failover over geographically distributed sites, you can use multiple data center replication over WAN. This entails setting up at least two master servers in each data center, and configuring the servers to be fully meshed over the WAN. This strategy prevents loss of service if any of the masters in the topology fail. Write operations must be routed to an alternative server if a writable server becomes unavailable.

6.3.3 Understanding the Use of Directory Proxy Server as Part of a Redundant Solution

You can use proxy servers to implement redundancy via several instances of proxy. This is yet another approach to provide highly available directory service.

Directory Proxy Server is designed to support high availability directory deployments. The proxy provides automatic load balancing as well as automatic failover and fail back among a set of replicated Directory Servers. Should one or more Directory Servers in the topology become unavailable, the load is proportionally redistributed among the remaining servers.

Directory Proxy Server actively monitors the Directory Servers to ensure that the servers are still online. The proxy also examines the status of each operation that is performed. Servers might not all be equivalent in throughput and performance. If a primary server becomes unavailable, traffic that is temporarily redirected to a secondary server is directed back to the primary server as soon as the primary server becomes available.

> **Note:**
>
> If you have distributed data, then you must manage multiple disconnected replication topologies, which makes administration more complex. In addition, Directory Proxy Server relies heavily on the proxy authorization control to manage user authorization. You must create a specific administrative user on each Directory Server that is involved in the distribution, and these administrative users must be granted proxy access control rights.

6.3.4 Understanding the Use of Application Isolation for High Availability

Directory Proxy Server can also be used to protect a replicated directory service from failure due to a faulty client application. To improve availability, a limited set of masters or replicas is assigned to each application.

Consider a scenario where a faulty application causes a server shutdown when the application performs a specific action. If the application fails over to each successive replica, a single problem with one application can result in failure of the entire replicated topology. To avoid such a scenario, you can restrict failover and load balancing of each application to a limited number of replicas. The potential failure is then limited to this set of replicas, and the impact of the failure on other applications is reduced.

6.3.5 Understanding How to Use the Replication Gateway for High Availability

The replication gateway is designed to provide a highly available deployment solution by allowing you to use redundant replication gateway servers for propagating changes made on disparate servers to the entire replication topology.

The replication gateway propagates changes between Oracle Directory Server Enterprise Edition and Oracle Unified Directory topologies. See Understanding the Role of the Replication Gateway.

6.4 Sample Topologies Using Redundancy for High Availability

When a failure occurs, sample topologies show redundancy and replication and provide continuous service.

For sample topologies that show how redundancy and replication can provide continued service when a failure occurs, see the following:

- Understanding Deployment Scenarios Using the Directory Server
- Understanding Deployments Using the Proxy Server
- Configuring Load Balancing Using OUDSM
- Replicating Global Index Catalogs

7

Understanding the Oracle Unified Directory Replication Model

It is essential to gain insight of the replication mechanism in Oracle Unified Directory to design robust and high available topologies.

> **Note:**
>
> The architectural topics described in this chapter are targeted at developers and at users who want to understand the internal replication mechanism. You do not have to read these topics to configure and use replication. For information about configuring and using replication, see Replicating Directory Data.

7.1 Overview of the Replication Architecture

You ideally need a replication setup for high-availability deployment and to enhance performance. Usually, there are multiple instances of Oracle Unified Directory Server in a viable network. Replication synchronizes the directory entries across these servers.

The following topics describe the replication architecture and various elements that comprise this architecture:

7.1.1 About Replication

Replication is built around a centralized publish-subscribe architecture. Each directory server communicates with a central service, and uses the central service to publish its own changes and to receive notification about changes on other directory servers. This central service is called the *replication service*.

Oracle Unified Directory uses a loosely consistent multi-master replication model, which means that all the directory servers within a replication topology can accept `read` and `write` operations.

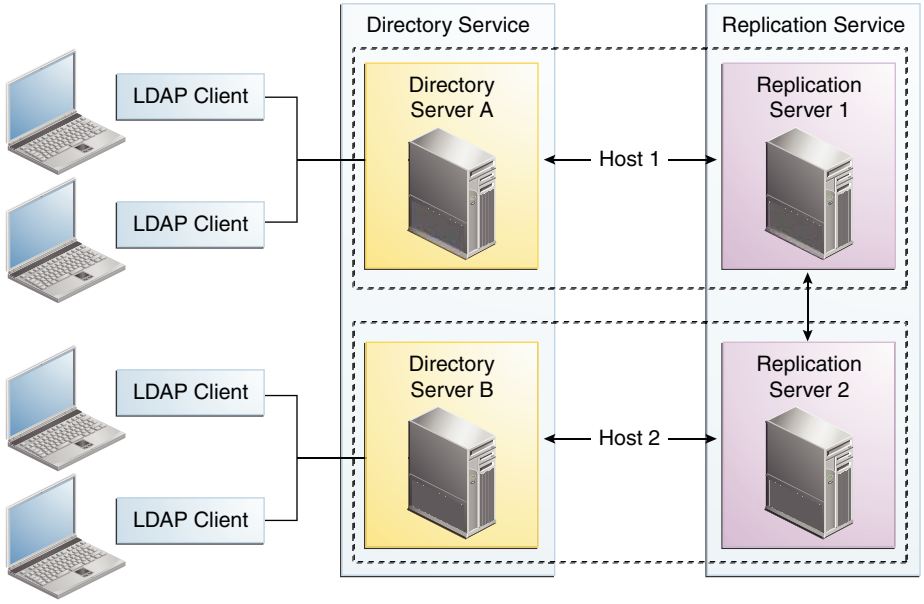
The replication service can be made highly available by using multiple server instances running on multiple hosts. Within the replication architecture, a server instance that provides the replication service is called a *replication server*. A server instance that provides the directory service is called a *directory server*.

The parties in a replication session authenticate to each other using SSL certificates. A connection is accepted if the certificate that is presented is in the ADS trust store. No access control or privileges are enforced.

7.1.2 Basic Replication Architecture

In a basic replication architecture, each directory server selects a single replication server and connects to it during startup. The directory server sends all changes that it processes to that replication server, and receives all changes from other servers in the topology through that replication server.

The basic replication architecture is shown in the following illustration.



> **Note:**
>
> In a replication architecture, each replication server is connected to every other replication server in the topology.

When a replication server receives a change from a directory server, the replication server forwards the change to all the other replication servers in the topology. These replication servers in turn forward the change to all the directory servers to which they are connected. When a replication server receives a change from another replication server, the replication server forwards the change to the directory servers to which it is connected, but not to other replication servers. A directory server never sends a change directly to another directory server. This architecture ensures that all changes are forwarded to all servers without requiring complex negotiation.

Every change is assigned a *change number* by the directory server that originally processed the change. The change number is used to identify the change throughout its processing. A replication server maintains changes in persistent storage so that older changes can be resent to directory servers that were not connected when the change occurred or that fell behind, becoming temporarily unable to receive all the changes at the time they were processed. See Replication Change Numbers.

The current update state of each directory server is maintained by keeping a record of the last changes that the directory server processed. When a directory server connects to a replication server, the replication server uses this record to determine the first change in the list of updates to send to the directory server.

Because multiple directory servers can process updates simultaneously, an update operation on one directory server can conflict with another update operation that affects the same entries on another directory server. Each directory server resolves conflicts when it replays operations from other directory servers, so that all directory server data eventually converges.

Conflicts can occur because of conflicting modify operations, called *modify conflicts*. Conflicts can also occur because of conflicting add, delete, or modRDN operations, called *naming conflicts*. To resolve conflicts in a coherent way, directory servers maintain a history of successive changes to each entry. This history is called *historical information*. Historical information is stored as an operational attribute inside the entry on which the changes occurred. See Overview of Historical Information and Conflict Resolution.

7.1.3 Replication Servers

Replication enables you to transmit copies of identical data across multiple servers. Replication servers propagates updates among a set of directory servers configured as replicas.

A replication server performs the following tasks:

- Manages connections from directory servers
- Connects to other replication servers
- Listens for connections from other replication servers
- Receives changes from directory servers
- Forwards changes to directory servers and other replication servers
- Saves changes to stable storage, which includes trimming older operations

Replication servers are different from directory servers. However, like directory servers, replication servers use a configuration file, and they can be configured, monitored online,

backed up, and restored. Replication servers are therefore always LDAP servers or JMX servers, even though replication servers do not store directory data.

When you configure a directory server instance for replication, a replication server is created automatically, unless you specify otherwise. The replication server and the directory server can run in the same JVM, or in separate JVMs.

In a small topology (up to four directory servers) it makes sense to configure each server to function as both a directory server and a replication server. In a large topology (more than twenty directory servers) it is advisable to separate the directory server and replication server instances into separate JVMs, and to limit the number of replication servers.

Between these two extremes, you can decided on the configuration that works best for your requirements. Having all servers functioning as both directory servers and replication servers is generally a simpler topology and easier to administer. Separating the directory servers and replication servers lowers the disk requirements of the directory server instances because they do not need to store a replication change log.

7.1.4 Replication Change Numbers

Change numbers uniquely identify changes that are made on an LDAP directory server. Change numbers also provide a consistent ordering of changes. The change number order is used to resolve conflicts and to determine the order in which forwarded changes should be replayed.

A *change number* consists of the following elements:

- **Time stamp, in milliseconds.** Time stamps are generated using the system clock. The change number is also generated such that each change number is always greater than all the change numbers that have already been processed by the server. Constantly increasing change numbers guarantees that operations that depend on previous operations are consistently replayed in the correct order. An example of an operation that depends on a previous operation is a modify operation that directly follows the add operation for that entry.
- **Sequence number.** A sequential number, increment for each change that occurs within the same millisecond.
- **Replica identifier.** A unique integer identifier that is assigned to each replica in a topology. (A *replication topology* is the set of all replicas of a given data set. For example, the replication topology for `example.com` might be all copies of the `dc=example,dc=com` suffix across a directory service.)

  The replica identifier ensures that two different servers do not assign the same identifier to two different changes. In a future directory server release, an algorithm might be used to assign replica identifiers automatically.

7.1.5 Replication Server State

When a directory server connects to a replication server, the replication server must determine how up to date the directory server data is before the replication server can send changes that the directory server has not yet seen. This "up to date" state of the directory server is called the *replication server state*.

A server might have missed relatively old changes from another remote server, yet might already have seen and processed more recent changes from a server that is close by. Server state is therefore maintained by recording the last change number processed by each replica, according to the replica identifier.

Because administrators can stop and restart servers, the server state must be saved to stable storage. Ideally saving the server state would be done after each local or replicated change is made. Saving information to the database after each change would add significant overhead, however. Server state is therefore kept in memory and saved to the database on a regular basis, and when the server is properly shut down.

A severe interruption to the server connection, such as a `kill` operation or a system failure, can cause the server to lose track of changes that have already been processed. This can result in the need to fix inconsistencies when the server restarts. For an explanation of how crash recovery is managed, see What is Directory Server Crashes?.

7.1.6 Operation Dependencies

Sometimes an operation cannot be replayed until another operation is complete. It is essential to gain an in depth understanding of such dependencies.

Consider a scenario, when an add operation is followed by a modify operation on the same entry, the server must wait for the add operation to be completed before starting the modify operation.

Such dependencies are quite rare and are generally necessary for a few operations only. Usually operations do not have dependencies, since they are modify operations. Therefore, in such cases, it is necessary to replay operations in parallel to obtain the best performance with multi-CPU servers.

The replication model is built on the assumption that operation dependencies are rare. The replication mechanism therefore always tries to replay operations in parallel, and only switches to processing operation dependencies if an operation fails to replay.

7.2 Understanding the Replication Mechanism

Oracle Unified Directory supports numerous replication capabilities. However, you must understand the mechanics involved in the replication process and how the different functionality are used.

This section consists of the following topics:

- Understanding Replication Initialization
- About Directory Server Change Processing
- Understanding Replication Server Selection
- Understanding Change Replay
- Understanding Auto Repair
- What is Directory Server Crashes?
- What is Replication Server Crashes?

7.2.1 Understanding Replication Initialization

Before a server can participate in a replicated topology, you must initialize that server with data. In other words, a complete data set must be copied onto the server.

Replication is automatic for data, but it has to be manually triggered for configuration. For information about the methods for initializing a server with data, see Initializing a Replicated Server With Data

Oracle Unified Directory configuration is specified in the file *instance-path*`/config/config.ldif`. This section lists the specific configuration attributes that you must replicate from the old instance to the new instance manually.

You can migrate the values of the following configuration attributes:

- Global configuration attributes, for instance writability mode, size and time limit, and so on.
- Security configuration attributes, for instance crypto manager, key manager, trust manager, ID mapping, and SASL.
- Connection handlers.
- Performance tuning attributes, for instance cache, threads, and other database configuration parameters.
- Replication configuration attributes.
- Password policy configuration attributes.
- Plug-In configuration attributes.
- Feature configuration attributes, for instance identity mapping, indexes, and so on.

7.2.2 About Directory Server Change Processing

When an update is performed on a directory server, replication code on the directory server performs numerous tasks. You must understand these tasks for a thorough insight.

The following tasks are triggered whenever any modification is initiated on a directory server:

- Assigns a change number
- Generates historical information
- Forwards the change to a replication server
- Updates the server state

Historical information is stored in the entry and must therefore be included in the operation before the server writes to the back end. The server uses the change number when generating historical information. The change number is therefore generated before the historical information. Both the change number and the historical information are performed as part of the pre-operation phase.

The operation is sent to the replication server before an acknowledgment for the update is sent to the client application that requested the operation. This ensures that a synchronous, assured replication mode can be implemented. See Understanding Assured Replication. The acknowledgment is therefore sent as part of the post-operation phase.

Changes are sent in the order defined by their change numbers. The correct order enables replication servers to ensure that all the changes are forwarded to other directory servers.

Because a directory server is multi-threaded, post-operation plug-ins can be called in a different order to pre-operation plug-ins, for the same operation. The replication code maintains a list of *pending changes*. This list includes changes that have started, and for which change numbers have already been generated, but that have not yet been sent to the replication server. Changes are added to the list of pending changes in the pre-operation phase. Changes are removed from the list when they are sent to the replication server. If a specific operation reaches the post-operation phase ahead of its change number-defined position, that operation waits until previous operations are sent before being sent to the replication server.

The server state is updated when the operation is sent to the replication server. See Replication Server State.

7.2.3 Understanding Replication Server Selection

When a directory server starts (or when the replication server to which it is connected is stopped), the directory server selects a suitable replication server for publishing and receiving changes.

The following sections describe how the replication server is selected and how the server load is balanced:

- About Replication Server Selection Algorithm
- Understanding Replication Server Load Balancing

7.2.3.1 About Replication Server Selection Algorithm

The directory server uses the following principles to select a suitable replication server:

- **Filtering.** To begin, the directory server creates a list of eligible replication servers, from all of the configured replication servers in the topology. The list is created based on the following criteria:
  1. Replication servers that have the same group ID (or geographic identifier) as the directory server.
  2. Replication servers that have the same generation ID (initial data set) as the directory server.
  3. Replication servers that include all of the latest updates that were generated from the directory server.
  4. Replication servers that run in the same virtual machine as the directory server.

  > **Note:**
  >
  > These criteria are listed in order of preference. So, for example, if a replication server has the same generation ID (criterion 2) as the directory server but does not have the same group ID (criterion 1), it will not be included in the list, unless no replication server in the topology has the same group ID as the directory server.

- **Load Balancing.** When the directory server has compiled a list of eligible replication servers, it selects a replication server in a manner that balances the load across all the replication servers in the topology. This selection is made in accordance with the *replication server weight* in the topology. See Understanding Replication Server Load Balancing.

7.2.3.2 Understanding Replication Server Load Balancing

In large topologies with several directory servers and several replication servers, it is more efficient to spread the directory servers out across the replication servers in a predefined manner. This approach is particularly important if the replication servers run on different types of machines, with different capabilities. If the estimated "global power" of the machines differs significantly from one replication server to another, it is useful to balance the load on the replication servers according to their power.

You can configure the proportional *weight* of a replication server so that the number of directory servers connecting to each replication server is balanced efficiently. Replication server weight is defined as an integer (`1..n`). Each replication server in a topology has a default weight of 1. This weight only has meaning in its comparison to the weights of other replication servers in the topology.

The replication server weight determines the proportion of the directory servers currently in the topology that should connect to this particular replication server. The replication server weight is configured as a fraction of the estimated global power of all the replication servers in the topology. For example, if replication server A is estimated to be twice as powerful as replication server B, the weight of replication server A should be twice the weight of replication server B.

The percentage of load of a particular replication server can be represented as ($^{n}/_{d}$) where *n* is the weight of the replication server and *d* is the sum of the weights of all the replication servers in the topology.

See Configuring the Replication Server Weight.

7.2.4 Understanding Change Replay

The replay of changes on replicated directory servers is efficient on multi-core and multi-CPU systems. On a directory server, multiple threads read the changes sent by the replication server.

Dependency information is used to decide whether an operation can be replayed immediately. The server checks the server state and the list of operations on which the current operation depends to determine whether those operations have been replayed. If the operations have not been replayed, the server puts the operation in a queue that holds dependency operations. If the operation can be replayed, the server builds an internal operation from information sent by replication servers. The server then runs the internal replay operation.

Internal replay operations built from the operations that are sent by a replication server can conflict with prior operations. Such internal operations cannot therefore always be replayed as if they were taking place on the original directory server. The server checks for conflicts when processing the *handleConflictResolution* phase.

In the majority of cases, the internal replay operations do not conflict with prior operations. In such cases, the handleConflictResolution phase does nothing. The replication code is therefore optimized to return quickly.

When a conflict does occur, the handleConflictResolution code takes the appropriate action to resolve the conflict. For modify conflicts, the handleConflictResolution code changes the modifications to retain the most recent changes.

When conflict resolution is handled, historical information is updated as for local operations. The operation can then be processed by the core server. Finally, at the end of the operation, the server state is updated.

After completing an operation, the server thread processing the operation checks whether an operation in the dependency queue was waiting for the current operation to complete. If so, that operation is eligible to be replayed, so the thread starts the replay process for the eligible operation. If not, the thread listens for further operations from the replication server.

7.2.5 Understanding Auto Repair

The auto repair mechanism is implemented as an LDAP application, and runs on the hosts that run replication servers. It primarily repairs inconsistent data.

Despite efforts to keep servers synchronized, directory servers can begin to show incoherent data. Typically, this occurs in the following circumstances:

- A disk error taints the stored data
- A memory error leads to an error in processing data
- A software bug leads to bad data or missing changes

In such cases, tracking and replaying changes is not sufficient to synchronize the incoherent data.

An automatic repair mechanism is provided, which leverages historical information inside entries to determine what the coherent data should be. The replication mechanism then repairs the data on directory servers where the data is bad or missing using auto repair application.

The auto repair application can run in different modes. Depending on the mode in which it is run, the auto repair application performs the following tasks:

- Repairs inconsistencies manifested as an error when the server was replaying modifications
- Repairs inconsistencies detected by the administrator
- Periodically scans directory entries to detect and repair inconsistencies

> **Note:**
>
> In the current directory server release, the auto repair mechanism must be run manually. See Detecting and Resolving Replication Inconsistencies.

7.2.6 What is Directory Server Crashes?

If a directory server crashes, its connection to the replication server is lost. Recent changes that the directory server has processed and committed to its database might not yet have been transmitted to any replication server.

When a directory server restarts, therefore, it must compare its state with the server state of the replication servers to which the directory server connects. If the directory server detects that changes are missing and not yet sent to a replication server, the directory server constructs fake operations from historical information. The directory server sends these fake operations to its replication server.

Because the local server state is not saved after each operation, the directory server cannot trust its saved server state after a crash. Instead, it recalculates its server update state, based on historical information.

7.2.7 What is Replication Server Crashes?

If a replication server crashes, directory servers connect to another replication server in the topology. The directory servers then check for and, if necessary, resend the missing changes.

7.3 Overview of Historical Information and Conflict Resolution

You need to learn how to handle historical information for resolving replication conflicts.

This section contains the following topics:

7.3.1 What is a Replication Conflict?

Replication conflicts occur in replication environment that allows concurrent updates to the same data at multiple sites. For instance, when two transactions that originate from two different sites update the same data at nearly the same time, a conflict can occur.

A conflict occurs when one or more entries are updated simultaneously on multiple servers and the changes are incompatible, or causes some interaction between the updates. Conflict occurs because no update operation is carried out simultaneously on every replica in the replication topology. Instead, updates are first processed on one server, then replicated to other servers.

The following example describes a conflict that occurs when an attribute is modified at the same time on two different directory servers.

Consider a topology with two read/write replicas. A modify operation changes the surname, `sn`, attribute of an entry to `Smith` on one server. Before the server that is processing the change can synchronize with the other server, the `sn` attribute value for that entry is replaced with the value `Jones` on the other server. Unless the conflict is managed, replication would replay the change (`Smith`) on the server that now contains the value `Jones`. At the same time, replication would replay the change (`Jones`) on the server that contains the value `Smith`. The servers would therefore end up with inconsistent values for the `sn` attribute on the modified entry.

The following list describes additional conflicts that can occur.

- An entry is deleted on one server while one of its attribute values is modified on another server.
- An entry is renamed on one server while one of its attribute values is remodified on another server.
- An entry is deleted and another entry with the same Distinguished Name (DN) is added on one server while one of its attribute values is modified on another server.
- A parent entry is deleted and a child of that entry is created on another server, either through an add operation or a rename operation.
- Two different entries with the same DN are added at the same time on two different servers.
- Two different values are used to replace a single-valued attribute on the same entry on different servers at the same time.

Conflicts that involve only modifications of the same entry are called *modify conflicts*. Conflicts that involve at least one operation other than modify are called *naming conflicts*.

All modify conflicts and the vast majority of naming conflicts can be solved automatically by replaying the operations in their order of occurrence. However, the following naming conflicts, which have very little chance of occurring, cannot be solved automatically.

- Two entries with the same DN are created at the same time on different servers, either by adding new entries or by renaming existing entries.

- A parent entry is deleted and a child of the parent entry is created at the same time. The child entry can be created either when a new entry is added or when an existing entry is renamed.

7.3.2 About Modify Conflict Resolution

Modify conflicts only occur with modification operations. You must device strategies to resolve such conflicts if they occur.

Operations are globally and logically ordered to determine the outcome of a given set of operations. Change numbers are used to define the order.

The replication conflict resolution functionality ensures that all servers eventually reach the same state, as if all operations were replayed everywhere in the order defined by the change numbers. This remains true even though changes might be replayed in a different order on different servers. In the modify conflict example with the `sn` values of `Smith` and `Jones`, described previously, assume that the value was set to `Jones` on the second server *after* it was set to `Smith` on the first server. The resulting attribute value should be `Jones` on both servers, even after the replace modification of `Smith` is replayed on the second server.

Historical information about each entry is retained to check whether a conflicting operation has already been played using a change number newer than that of a current conflicting operation. For each modify operation, historical information is used, first to check if there is a conflict, and, if there is a conflict, to determine the correct result of the operation.

When a modify conflict occurs, the server determines whether the current attribute values must be retained or whether the modification must be applied. The current attribute values alone are not sufficient to make this assessment. The server also determines when (at which change number) prior modifications were made. Historical information therefore includes the following elements:

- The date when the attribute was last deleted
- The date when a given value was last added
- The date when a given value was last deleted

When an attribute is deleted or fully replaced, older information is no longer relevant. At that point the older historical information is removed.

Historical information undergoes the following processing:

- Saved in the `ds-sync-hist` attribute (can be viewed only by an administrator)
- Updated (but not used) for normal operations
- Updated and used for replicated operations

Conflict resolution is carried out when operations are replayed, after the pre-operation during the `handleConflictResolution` phase.

Conflict resolution is carried out by changing the `List<Modification>` field of the `modifyOperation` to match the actual modifications required on the user attributes of the entry, and to change the `ds-sync-hist` attribute that is used to store historical information.

7.3.3 About Naming Conflict Resolution

Naming conflicts only happen for replayed operations. You need to identify how to resolve these conflicts using an appropriate strategy.

The server uses the following methods to resolve naming conflicts:

- Uses unique IDs to identify entries, including entries that have been renamed
- Tries to replay each operation first and only takes action if a conflict occurs
- Checks during the pre-operation phase for conflicts that cannot be detected when operations are replayed
- Retains no *tombstone entries*, which are entries that have been marked for deletion but not yet removed

Because directory entries can be renamed, the DN is not an immutable value of the entry. DNs cannot therefore be used to identify the entry for replication purposes. A unique and immutable identifier is therefore generated when an entry is created, and added as an operational attribute of the entry. This unique ID is used, instead of the DN, to identify the entry in changes that are sent between directory servers and replication servers.

A replication context is attached to the operation. The replication context stores private replication information such as change number, entry ID, and parent entry ID that is required to solve the conflict.

7.3.4 Understanding How to Purge Historical Information

Historical information is stored in the server database. Historical information therefore consumes space, I/O bandwidth, and cache efficiency. Historical information can be removed as soon as more recent changes have been seen from all the other servers in the topology.

Historical information is purged in the following ways:

- When a new change is performed on the entry.
- By a purge process that can be triggered at regular intervals. The purge process saves space, at the cost of more CPU for processing the purge. The purge process is therefore configurable. See Configuring Replication Purge Delay.

7.4 Overview of Schema Replication

Schema replication is described to the users by schema replication architecture. You must have an in-depth understanding of the architecture to implement schema replication.

This section contains the following topics:

- About Schema Replication.
- Schema Replication Architecture.

7.4.1 About Schema Replication

Schema describe the entries that can be stored in a directory server. Schema management is a core feature of the directory service. Replication is also a central feature of the directory service and is essential to a scalable, highly available service.

Any changes made to the schema of an individual directory server must therefore be replicated on all the directory servers that contribute to the same service.

Schema replication occurs when the schema is modified in any of the following ways:

- By modifying the `cn=schema` suffix when the server is online
- By using a dedicated task to perform dynamic schema updates by means of a file when the server is online

- By modifying the underlying back-end files directly when the server is offline

Generally, schema modifications occur only when deploying new applications or new types of data. The rate of change for schema is therefore low. For this reason, the schema replication implementation favors simplicity over scalability.

Schema replication is enabled by default. In certain specific cases, it might be necessary to have different schema on different directory servers, even when the servers share all or part of their data. In such cases you can disable schema replication, or specify a restricted list of servers that participate in schema replication. See Configuring Schema Replication

7.4.2 Schema Replication Architecture

The schema replication architecture relies heavily on the general replication architecture. Therefore, it is recommended that you have a thorough understanding of the general replication architecture before reading this section.

Directory servers notify replication servers about any changes to their local schema. As with data replication, the replication servers propagate schema changes to other replication servers, which in turn replay the changes on the other directory servers in the topology. See Overview of the Replication Architecture.

Schema replication shares the same replication configuration used for any subtree:

```
dn: cn=example,cn=domains,cn=Multimaster Synchronization,\
  cn=Synchronization Providers,cn=config
objectClass: top
objectClass: ds-cfg-replication-domain
cn: example
ds-cfg-base-dn: cn=schema
ds-cfg-replication-server: <server1>:<port1>
ds-cfg-replication-server: <server2>:<port2>
ds-cfg-server-id: <unique-server-id>
```

Schema replication differs from data replication in the following ways:

- **Entry Unique ID.** A unique ID is required for data replication because entries can be renamed or deleted.

  In the schema, there is only one entry and that entry cannot be deleted or renamed. The unique ID used for the schema entry is therefore the DN of the schema entry.

- **Historical information.** Historical information is used to save a history of relevant modifications to an entry. This information makes it possible to solve modification conflicts.

  For schema replication, the only possible operations are adding values and deleting values. Historical information is therefore not maintained for modifications to the schema.

- **Persistent server state.** When a directory server starts up, the replication plug-in establishes a connection with a replication server. The replication server looks for changes in its change log and sends any changes that have not yet been applied to the directory server.

  To know where to start in the change log, the replication plug-in stores information that is persistent across server stop and start operations. This persistent information is stored in the replication `base-dn` entry.

  The schema back end allows the specific operational attribute used to store the persistent state, `ds-sync-state`, to be modified.

7.5 Overview of Replication Status

A *replication domain* is a directory server that contains data. Each replicated domain in a replicated topology has a certain *replication status*. The replication status is determined by the replication domain connections within the topology, and by how up-to-date the replication domain is based on the changes that have occurred throughout the topology.

Knowledge of a domain's replication status enables a replicated topology to do the following:

- Manage certain aspects of assured replication
- Enable certain administrative tasks
- Administer and monitor replication effectively

See Monitoring a Replicated Topology.

The following sections outline the different statuses that a replicated domain can have:

- Replication Status Definitions
- What is Degraded Status?
- Understanding Full Update Status and Bad Generation ID Status

7.5.1 Replication Status Definitions

There are multiple status values that you can set for directory servers that contain data in a replication domain.

The status can be one of the following:

- **Normal.** The connection to a replication server is established with the correct data set. Replication is working. If assured mode is used, then acknowledgments from this directory server are sent.
- **Late.** The connection to a replication server is established with the correct data set. Replication is marked Late when the number of missing changes for the directory server exceeds the threshold defined in the replication server configuration. When the number of changes goes below this threshold, the status will go back to Normal.
- **Full Update.** The connection to a replication server is established and a new data set is received from this connection (online import), to initialize the local back end.
- **Bad Data Set.** The connection to a replication server is established with a data set that is different from the rest of the topology. Replication is not working. Either the other directory servers of the topology should be initialized with a compatible data set, or this server should be initialized with another data set that is compatible with the other servers.
- **Not Connected.** The directory server is not connected to any replication server.
- **Unknown.** The status cannot be determined. This occurs mainly when the server is down or unreachable but it is referenced in the monitoring of another server.
- **Invalid.** This is for internal use. If the directory server changes its state and the transition is impossible according to state machine, then `INVALID_STATUS` is returned.

7.5.2 What is Degraded Status?

A directory server that is slow in replaying changes is assigned a `DEGRADED_STATUS`.

The stage at which the server is regarded as "too slow" is defined by the *degraded status threshold* and is configurable, based on the number of updates queued in the replication server for that directory server.

When the degraded status threshold is reached, the directory server assumes a degraded status and is considered to be unable to send acknowledgments in time. A server with this status can have an impact on assured replication, as replication servers no longer wait for an acknowledgment from this server before returning their own acknowledgments.

7.5.3 Understanding Full Update Status and Bad Generation ID Status

A directory server can change status depending on the type of task performed by the administrator on the topology.

Apart from being assigned a degraded status, a directory server can be assigned another status depending on the following tasks performed on the topology:

- **Full update.** When a replicated domain is initialized online from another server in the topology, the directory server status for that domain changes to `FULL_UPDATE_STATUS`. When the full update has completed, the directory server reinitializes its connection to the topology, and the status is reset to `NORMAL_STATUS`.
- **Local import or restore.** When a replicated domain is reinitialized by using a local import or restore procedure, the directory server status for that domain changes to `NOT_CONNECTED_STATUS`.
- **Resetting the generation ID.** If a replicated domain connects to a replication server with a generation ID that is different from its own, the domain is assigned a `BAD_GEN_ID` status. A domain can also be assigned this status if a reconnection occurs after a full online update, a local import, or a restore with a set of data that has a different generation ID to that of the replication server.

  In addition, you might need to reset the generation ID of all the replication servers in the topology by running the reset generation ID task on the directory server. This causes all the replication servers in the topology to have a different ID to the ID of the directory servers to which they are connected. In this case, the directory servers are assigned a `BAD_GEN_ID` status.

7.6 About Replication Groups

Replication groups are designed to support multi-data center deployments and disaster recovery scenarios. Replication groups are defined by a group ID. A group ID is a unique number that is assigned to a replicated domain on a directory server (one group ID per replicated domain). A *group ID* is also assigned to a replication server (one group ID for the whole replication server).

Group IDs determine how a directory server domain connects to an available replication server. From the list of configured replication servers, a directory server first tries to connect to a replication server that has the same group ID as that of the directory server. If no replication server with a compatible group ID is available, the directory server connects to a replication server with a different group ID. The next section describes this selection process in greater detail. See Configuring Replication Groups.

> **Note:**
>
> Assured replication does not cross group boundaries. See Understanding Assured Replication.

7.7 Understanding Assured Replication

Assured replication is a method of making regular replication work in a more synchronized manner. The topics in this section describe how assured replication works from an architectural perspective.

> **Note:**
>
> Before you read the following sections, it is recommended that you should have an understanding of basic replication concepts. You must know what a replication server is, as opposed to a directory server, and have an understanding of how replication servers work in a replicated topology. If this is not the case, read at least the Overview of the Replication Architecture to obtain an understanding of how regular replication works in the directory server.

The following sections describe the implementation of assured replication:

- Need for Assured Replication
- Supported Assured Replication Modes Configuration
- Understanding Assured Replication Connection Algorithm
- Understanding Assured Replication and Replication Status
- Understanding Assured Replication Monitoring

7.7.1 Need for Assured Replication

Assured Replication ensures enhanced consistency of data between replicas. In assured replication the response to an LDAP update is delayed until the change is received or applied by other servers. This assures that the change is not lost even if the server receiving the change crashes.

In a standard replicated topology, changes are replayed to other replicated servers in a "best effort" mode. A change made on an LDAP server is replayed on the other servers in the topology as soon as possible, but in an unsynchronized manner. This is convenient for performance but does not ensure that a change has been propagated to other servers when the initial LDAP client call is finished.

In some deployments this might be acceptable, that is, the time period between the change on the first server and the replay on peer servers might be short enough to fulfill the requirements of the deployment. For example, an international organization might store employee user accounts in a replicated topology across various geographical locations. If a new employee is hired and a new account is created for him on one LDAP server in a specific location, it might be acceptable that the replay of the creation occurs in other LDAP servers a few milliseconds after the LDAP client call terminates. The user is unlikely to perform a host login that would access one of the other LDAP servers in the same second that the user account is created.

However, there might be cases in which more synchronization is required from the replication process. If a specific host fails, it might be imperative that any changes made on that host have been propagated elsewhere in the topology. In addition, the deployment might require assurance that once the LDAP client call of a change is returned by a server, all of the peer servers in the topology have received that change. Any other clients that read the entry from anywhere in the topology would be sure to obtain the modification.

Assured replication is a method of making regular replication work in a more synchronized manner. The topics in this section describe how assured replication works, from an architectural perspective. For information about configuring assured replication, see Configuring Assured Replication.

7.7.2 Supported Assured Replication Modes Configuration

The directory server currently supports several different kinds of assured replication modes, depending on the level of synchronization that is required, the goal of the replicated topology, and the acceptable performance impact. This section provides an in-depth coverage of the different assured replication modes.

This section contains the following topics:

- Example of Using Safe Data Mode
- Example of Using Safe Data Level = 1.
- Example of Using Safe Data Level = 2 (RS and DS on Different Hosts).
- Example of Using Safe Data Level = 2 (RS and DS on Same Host).
- Example of Using Safe Read Mode
- Understanding Safe Read Mode and Replication Groups
- Example of Using Safe Read Mode in a Single Data Center With One Group.
- Example of Using Safe Read Mode in a Single Data Center With More Than One Group.
- Example of Using Safe Read Mode in a Multi-Data Center Deployment.

7.7.2.1 Example of Using Safe Data Mode

In safe data mode, any change is propagated to a specified number of servers in the topology before the LDAP client call returns. If the LDAP server on which the change is made fails, it is guaranteed that the change is propagated to at least the specified number of servers.

This specified number of servers (N) defines the *safe data level*. The safe data level is based on acknowledgments from the replication servers only. In other words, an update message that is sent from an LDAP server must be acknowledged by at least N (N>=1) replication servers before the LDAP client call that initiated the update returns.

The higher the safe data level, the greater the number of machines that are assured to have the update and, consequently, the more reliable the data. However, as the safe data level increases, the overall performance decreases because additional acknowledgments are required before the LDAP client call returns.

The safe data level functions in best effort mode. That is, if the safe data level is set to 3 and there are temporarily only two replication servers available in the topology, an acknowledgment from the third (unavailable) replication server will not be expected until this server is available again.

Safe data mode is affected by the use of *replication groups*. Because assured replication does not cross group boundaries, a replication server with a group ID of 1 waits for an

acknowledgment from other replication servers with the same group ID but not for acknowledgments from replication servers with a different group ID. For more information, see About Replication Groups.

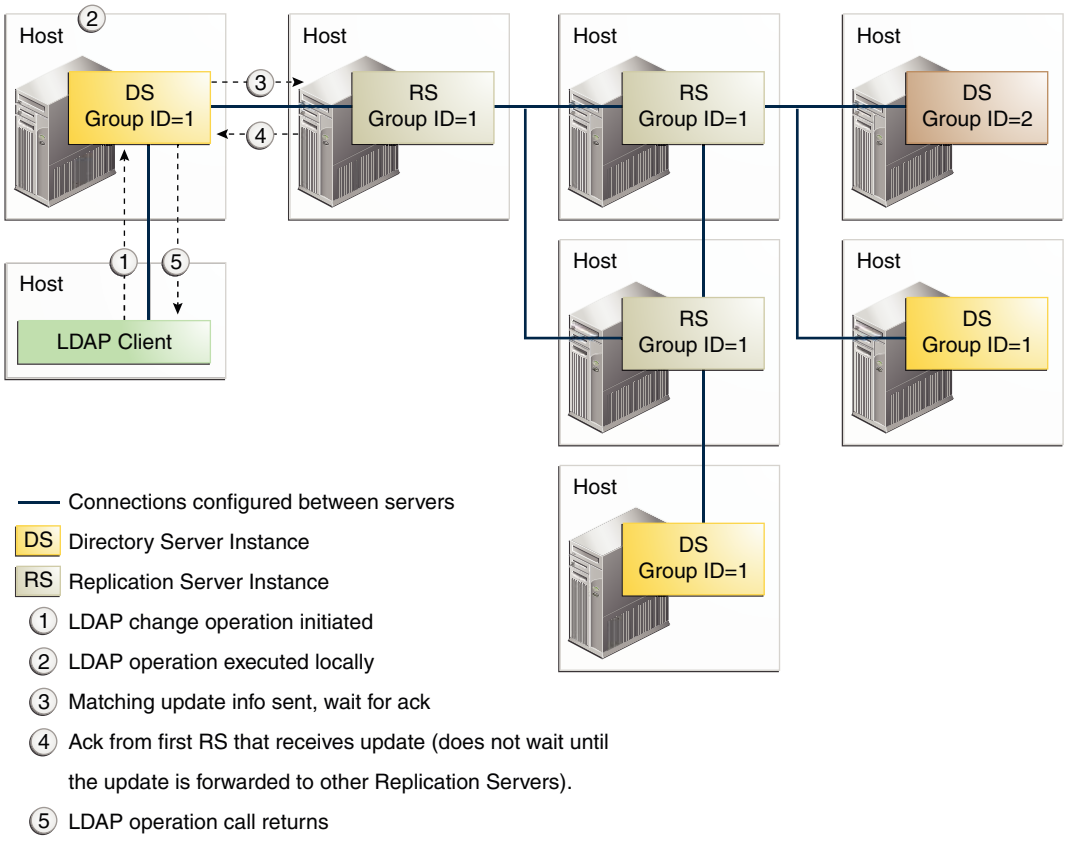
> **Note:**
>
> In the current replication implementation, the `setup` and `dsreplication` commands support only a scenario in which the main replication server is physically located in the same host as the LDAP server (that is, on the same machine). However, the fundamental replication design is to support deployments where the replication servers run on separate machines, to increase reliability.
>
> Such deployments can currently be configured only by using the `dsconfig` command and are not supported by the `setup` and `dsreplication` commands. However, these deployments provide better failover and availability. In such deployments, if the safe data level is set to 1 (acknowledgment of only one replication server is expected), this replication server *must* run on a separate machine to the LDAP server.

7.7.2.2 Example of Using Safe Data Level = 1

Setting the safe data level to 1 ensures that the first replication server returns an acknowledgment to the directory server immediately after receiving the update.

The replication server does not wait for acknowledgments from other replication servers in the topology. The modification is guaranteed to exist on one additional server (other than the directory server on which the change was made).

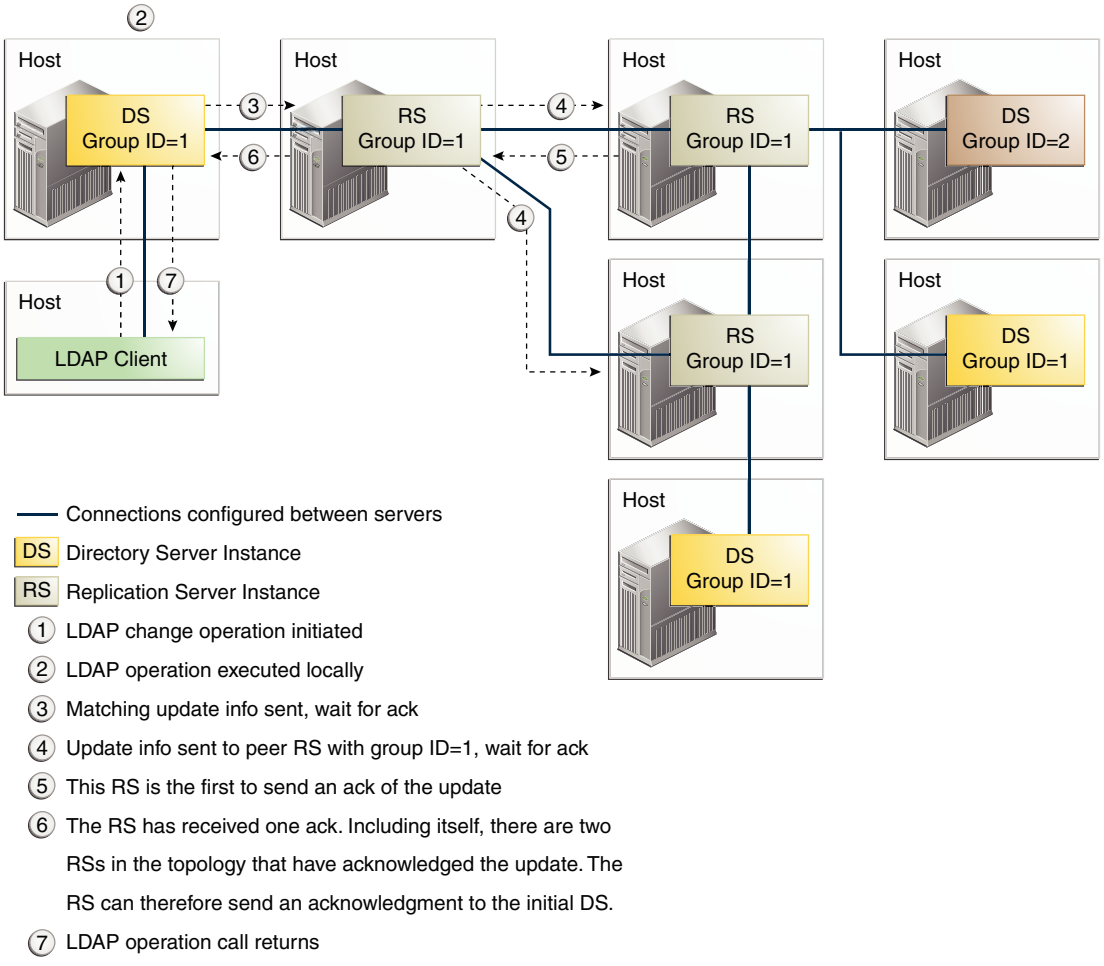
This example can only be configured with `dsconfig` and is not yet supported by the `setup` or `dsreplication` commands.



7.7.2.3 Example of Using Safe Data Level = 2 (RS and DS on Different Hosts)

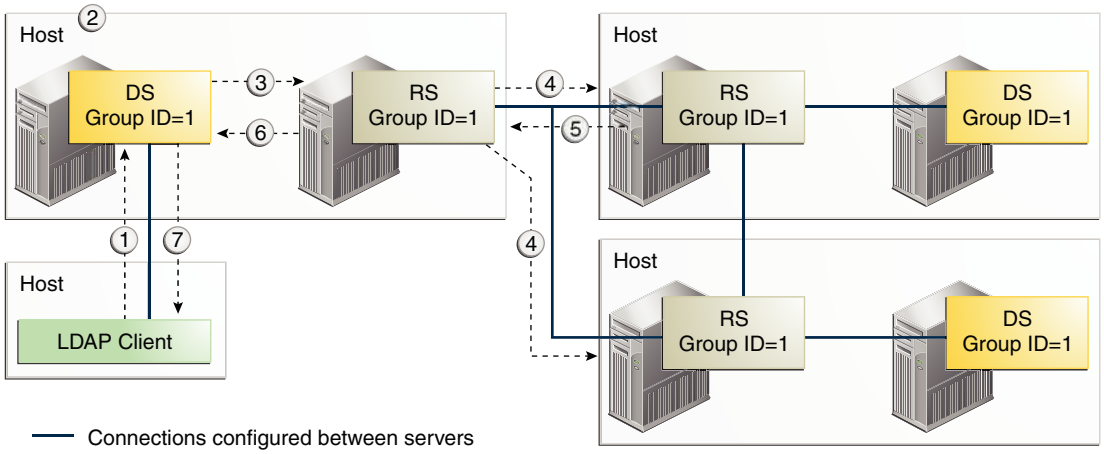
Setting the safe data level to 2 ensures that the first replication server will wait for an acknowledgment from one peer replication server before returning an acknowledgment to the directory server. The modification is guaranteed to exist on two additional servers (other than the directory server on which the change was made).

This example can only be configured with `dsconfig` and is not yet supported by the `setup` or `dsreplication` commands.



7.7.2.4 Example of Using Safe Data Level = 2 (RS and DS on Same Host)

In the current replication implementation, the `setup` and `dsreplication` commands only support configurations in which the replication is on the same machine as the directory server. With this implementation, if you want to ensure that a change is sent to at least one additional host, then you must set the safe data level to 2.



Connections configured between servers

DS Directory Server Instance

RS Replication Server Instance

1. LDAP change operation initiated
2. LDAP operation executed locally
3. Matching update info sent, wait for ack
4. Update info sent to peer RS with group ID=1, wait for ack
5. This RS is the first to send an ack of the update
6. The RS has received one ack. Including itself, there are two RSs in the topology that have acknowledged the update. The RS can therefore send an acknowledgment to the initial DS.
7. LDAP operation call returns

7.7.2.5 Example of Using Safe Read Mode

Safe read mode ensures that any modification made on a specific directory server has been replayed to all other directory servers within the topology before the LDAP call returns. In this mode, if another LDAP client performs a read operation on another directory server in the topology, that client is assured of reading the modification that has just been performed. Safe read mode is the most synchronized manner in which you can configure replication. However, this mode also has the biggest performance impact in terms of write time.

Safe read mode is based on acknowledgments from the LDAP servers rather than the replication servers in a topology. When a modification is made on a directory server, the update is sent to the corresponding replication server. The replication server then forwards the update to the other replication servers in the topology. These replication servers wait for acknowledgment of the modification being replayed on all the directory servers to which the modification is forwarded. When the modification has been replayed on all directory servers in the topology, the replication servers send their acknowledgment back to the first replication server, which in turn sends an acknowledgment to the original directory server.

The first replication server also waits for an acknowledgment from any other directory servers that are directly connected to it before sending the acknowledgment to the original directory server. Only when the original directory server has received an acknowledgment from its replication server does it finally return the end of the operation call to the LDAP client.

At this point, all directory servers in the topology contain the modification. If an LDAP client reads the data from any directory server, it is therefore certain of obtaining the modification.

7.7.2.6 Understanding Safe Read Mode and Replication Groups

Replication groups support multi-data center replication and high availability. For more information about replication groups, see About Replication Groups. In the context of assured replication, replication groups enable a set of directory servers to work together in safe read mode. All directory servers that work together in a synchronized manner require the same group ID. This group ID should also be assigned to all the replication servers working in the synchronized topology. Assured replication does not cross group boundaries.

When a change occurs on a directory server with certain group ID (N), the LDAP call is not returned before every other directory server with group ID N has returned an acknowledgment of the change.

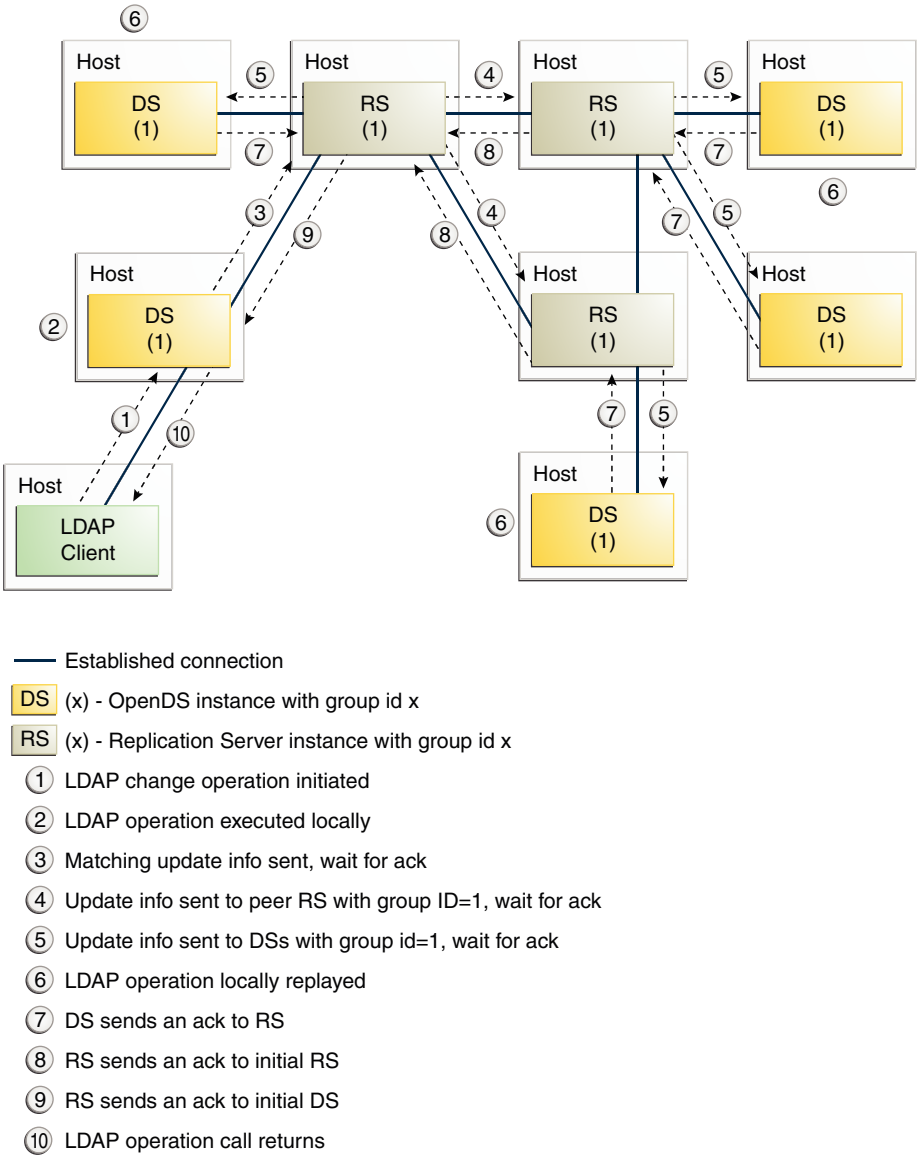
The use of replication groups provides more flexibility in a replicated topology that uses safe read mode.

- In a single data center deployment, you can define a subset of the directory servers that should be fully synchronized. Only the directory servers with the same group ID will wait for an acknowledgment from their peers with the same group ID. All the replication servers will have the same group ID.
- In a multi-data center deployment, you can specify that all the directory servers within a single data center are fully synchronized. A directory servers will wait for acknowledgment only from its peers located in the same data center before returning an LDAP call. Acknowledgment is expected only if the directory server is connected to a replication server with the same group ID.

7.7.2.7 Example of Using Safe Read Mode in a Single Data Center With One Group

The following illustration shows a deployment in which all nodes are in the same data center and are part of the same replication group. Each directory server and replication server has the same group ID. Any modification must be replayed on every directory server in the topology before an LDAP client call returns. Any subsequent LDAP read operation on any directory server in the topology is assured of reading the modification.

Such a scenario might be convenient, for example, if there is an LDAP load balancer in front of the replicated directory server pool. Because it is impossible to determine the directory server to which the load balancer will redirect an LDAP modification, a subsequent read operation is not necessarily routed to the directory server on which the modification was made. In this case, it is imperative that the change is made on all servers in the topology before the LDAP client call is returned.

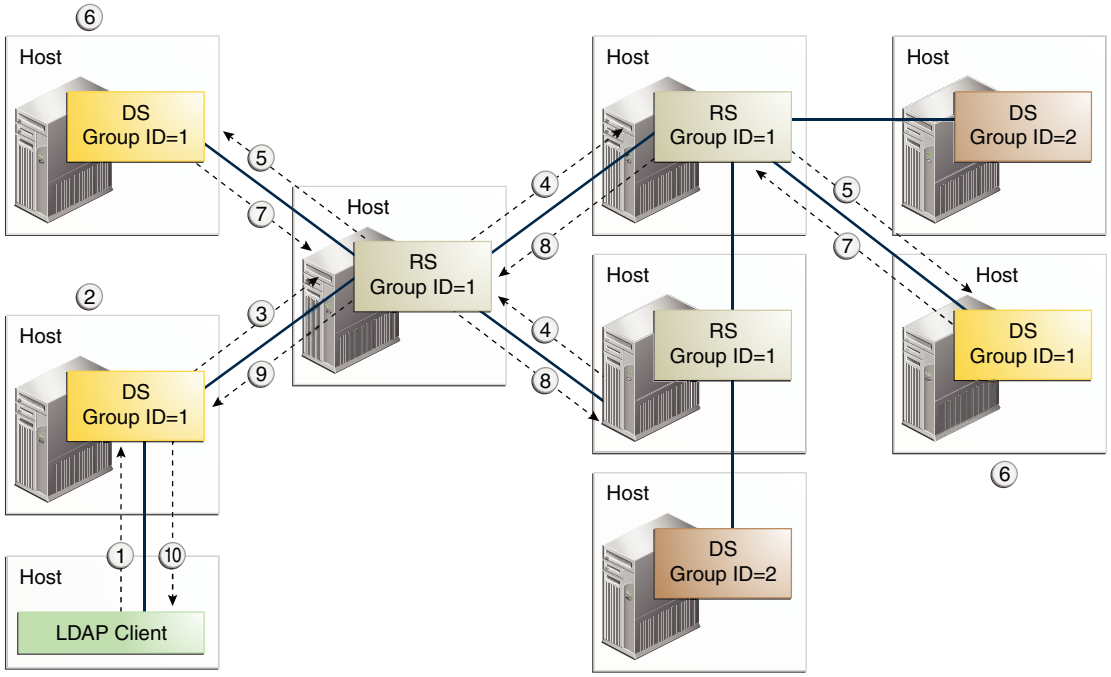


7.7.2.8 Example of Using Safe Read Mode in a Single Data Center With More Than One Group

The following illustration shows a deployment in which all nodes are in the same data center but in which assured replication is configured on only a subset of the directory servers. This subset of servers constitutes a replication group, and each server is assigned the same group ID (1). When a change is made on one of the directory servers in the replication group, an acknowledgment must be received from all the directory servers in the group before the initial LDAP call is returned to the client. The remaining directory servers in the topology will still replay the change, but their acknowledgment is not required before the LDAP call is returned. If a change made on one of the servers outside of the group, no acknowledgment from other directory servers is required before the LDAP call is returned to the client.

In this example, the replication server that is connected to directory servers outside of the replication group is still assigned a group ID of 1. This configuration ensures failover if another replication server is offline. In this case, if a directory server within the replication group

connects to this particular replication server, assured replication must still work. For the purpose of failover, any replication server must be assigned the same group ID if there is a chance that a directory server within the group might connect to it at some stage.



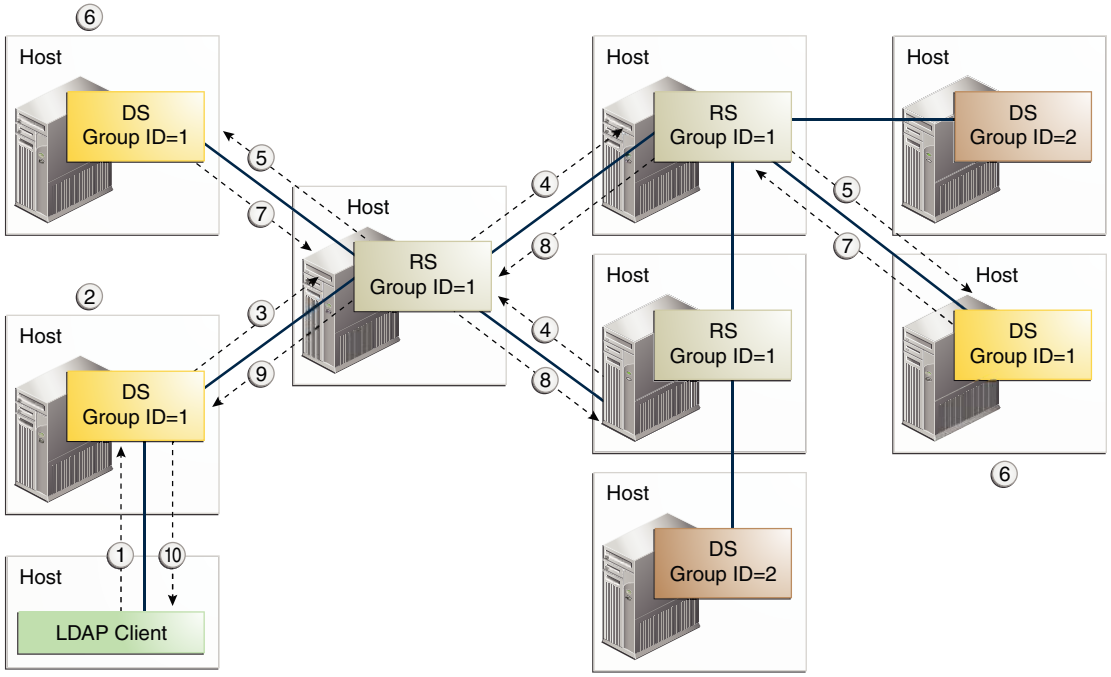
- Connections configured between servers
- (1) LDAP change operation initiated
- (2) LDAP operation executed locally
- (3) Matching update info sent, wait for ack
- (4) Update info sent to peer RS with group ID=1, wait for ack
- (5) Update info sent to peer DS with group ID=1, wait for ack
- (6) LDAP operation replayed locally
- (7) DS sends ack back to RS
- (8) RS sends ack initial RS
- (9) Initial RS sends ack to initial DS
- (10) LDAP operation call returns

7.7.2.9 Example of Using Safe Read Mode in a Multi-Data Center Deployment

The following illustration shows a deployment with two data centers (in different geographical locations). Each data center has safe read mode configured locally within the data center. All of the directory servers and the replication servers within the same data center are assigned the same group ID (1 for the first data center and 2 for the second data center). The directory servers within the same data center operate in a more tightly consistent synchronized manner. Any change made on a directory server must be replayed and acknowledged from all directory servers within that data center before the LDAP client call returns.

In this example, data is synchronized between the two data centers, but any change made on a specific directory server is immediately visible on all other directory servers within the same

data center. This scenario is convenient if there is an LDAP load balancer in front of the directory servers of a data center. The performance impact in terms of writes is not too great because no acknowledgments are requested from the servers of the remote data center.



Connections configured between servers

1. LDAP change operation initiated
2. LDAP operation executed locally
3. Matching update info sent, wait for ack
4. Update info sent to peer RS with group ID=1, wait for ack
5. Update info sent to peer DS with group ID=1, wait for ack
6. LDAP operation replayed locally
7. DS sends ack back to RS
8. RS sends ack initial RS
9. Initial RS sends ack to initial DS
10. LDAP operation call returns

The group ID of the replication server is important in this scenario. If a change arrives from a directory server with group ID N, the replication server compares N with its own group ID and takes the following action:

- If the replication server has the same group ID (N), it forwards the change to all the replication servers and directory servers to which it is directly connected. However, it waits for an acknowledgment only from the servers with the same group ID (N) before sending its own acknowledgment back to the original directory server.
- If the replication server has a different group ID, it forwards the change to all the replication servers and directory servers but does not wait for any acknowledgment.

7.7.3 Understanding Assured Replication Connection Algorithm

To implement the different scenarios of assured replication, the directory server in a topology makes use of an algorithm to select the replication server to which that directory server should connect with.

The algorithm that the directory server implements to select the replication server is as follows:

1. Connect to each replication server in the list of configured replication servers and obtain its server state and group ID.
2. From the list of replication servers that are up to date with the changes on the directory server, and that have same group ID as the directory server, select the one that has the most updates from other directory servers in the topology. If no replication server exists with the same group ID as the directory server, select the replication server that is most up to date.

This algorithm ensures that a higher priority is given to replication servers with the same group ID as the directory server's group ID. A directory server will therefore favor a replication server located in its own data center.

Connecting to a replication server with the same group ID (in the same data center) provides the safe read mode functionality. Connecting to a replication server with a different group ID provides failover to another data center (if all the replication servers in the local data center fail). In this case, safe read mode is disabled as no acknowledgment is requested when sending update messages to replication servers with a different group ID. Replication continues, but in degraded mode (that is, the safe read mode requested at configuration time is not applied.)

To return replication to normal, a directory server periodically polls the configuration list for the arrival of replication servers with the same group ID as its own. If the directory server detects that a replication server with its own group ID is available, it disconnects from the current replication server (with a different group ID), and reconnects to the recovered replication server with the same group ID. Safe read mode is thus re-enabled and replication returns to the mode in which it was configured.

7.7.4 Understanding Assured Replication and Replication Status

You can set the replication status while implementing assured replication to different values depending on the state of the directory server in the replication topology.

When a replication server detects that a directory server is out of sync regarding the overall updates made in the topology, that directory server is said to have a *degraded status*. A directory server that is out of sync is unlikely to be able to send the expected acknowledgments in time for the replication server to avoid a time-out situation. The server therefore has a degraded status until it has an acceptable level of updates. With a degraded status, a directory server is no longer expected to send acknowledgments to the replication server, until it returns to having a *normal status*.

Because a directory server with a degraded status cannot send acknowledgments, the synchronization of an LDAP operation in safe read mode cannot be assured. In other words, data read from this directory server might not contain the modifications made on another directory server in the topology.

For more information, see Replication Status Definitions.

7.7.5 Understanding Assured Replication Monitoring

The assured replication mechanism includes several attributes defined to monitor how well the mechanism is working.

This section describes the monitoring attributes defined on the directory servers and on the replication servers in a topology.

On a directory server, the attributes are located under the monitor entry for that replicated DN. For example, monitoring information related to the replicated domain `dc=example,dc=com` is located under the monitoring entry `cn=Replication Domain,dc=example,dc=com,server-id,cn=monitor`.

On a replication server, the monitoring information related to assured replication is on a per connection basis. Monitoring attributes are found in the monitoring entry of a directory server or replication server that is connected to the current replication server. For example, on a particular replication server, the monitoring information related to a connected directory server would be under the monitoring entry `cn=Directory Server dc=example,dc=com ds-host,server-id,cn=monitor`. The monitoring information related to a connected replication server would be under the monitoring entry `cn=Remote Replication Server dc=example,dc=com repl-server-host:repl-port,server-id,cn=monitor`.

Table 7-1 provides a list of attributes used for a directory server.

Table 7-1 Monitoring Attributes on the Directory Server

Attribute Name	Attribute Type	Purpose
`assured-sr-sent-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode
`assured-sr-acknowledged-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode, that have been successfully acknowledged
`assured-sr-not-acknowledged-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode, that have not been successfully acknowledged (either because of timeout, wrong status, or error at replay)
`assured-sr-timeout-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode, that have not been successfully acknowledged because of timeout
`assured-sr-wrong-status-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode, that have not been successfully acknowledged because of wrong status
`assured-sr-replay-error-updates`	Integer (0..N)	Number of updates sent in assured replication, safe read mode, that have not been successfully acknowledged because of replay error

Table 7-1 (Cont.) Monitoring Attributes on the Directory Server

Attribute Name	Attribute Type	Purpose
`assured-sr-server-not-acknowledged-updates`	String	Multiple values allowed: number of updates sent in assured replication, safe read mode, that have not been successfully acknowledged (either because of timeout, wrong status or error at replay) for a particular server (directory server or replication server). String format: *server id*:*number of failed updates*
`assured-sr-received-updates`	Integer (0..N)	Number of updates received in assured replication, safe read mode
`assured-sr-received-updates-acked`	Integer (0..N)	Number of updates received in assured replication, safe read mode that have been acknowledged without errors
`assured-sr-received-updates-not-acked`	Integer (0..N)	Number of updates received in assured replication, safe read mode, that have been acknowledged with errors
`assured-sd-sent-updates`	Integer (0..N)	Number of updates sent in assured replication, safe data mode
`assured-sd-acknowledged-updates`	Integer (0..N)	Number of updates sent in assured replication, safe data mode, that have been successfully acknowledged
`assured-sd-timeout-updates`	Integer (0..N)	Number of updates sent in assured replication, safe data mode, that have not been successfully acknowledged because of timeout
`assured-sd-server-timeout-updates`	String	Multiple values allowed: number of updates sent in assured replication, safe data mode, that have not been successfully acknowledged (because of timeout) for a particular RS. String format: *server id*:*number of failed updates*

Table 7-2 lists the monitoring attributes used for a replication server.

Table 7-2 Monitoring Attributes on the Replication Server

Attribute Name	Attribute Type	Purpose
`assured-sr-received-updates`	Integer (0..N)	Number of updates received from the remote server in assured replication, safe read mode
`assured-sr-received-updates-timeout`	Integer (0..N)	Number of updates received from the remote server in assure replication, safe read mode, that timed out when forwarding them
`assured-sr-sent-updates`	Integer (0..N)	Number of updates sent to the remote server in assured replication, safe read mode

Table 7-2 (Cont.) Monitoring Attributes on the Replication Server

Attribute Name	Attribute Type	Purpose
`assured-sr-sent-updates-timeout`	Integer (0..N)	Number of updates sent to the remote server in assured replication, safe read mode, that timed out
`assured-sd-received-updates`	Integer (0..N)	Number of updates received from the remote server in Assured Replication, Safe Data
`assured-sd-received-updates-timeout`	Integer (0..N)	Number of updates received from the remote server in assured replication, safe date mode, that timed out when forwarding them. This attribute is meaningless if the remote server is a replication server.
`assured-sd-sent-updates`	Integer (0..N)	Number of updates sent to the remote server in assured replication, safe data mode. This attribute is meaningless if the remote server is a directory server.
`assured-sd-sent-updates-timeout`	Integer (0..N)	Number of updates sent to the remote server in assured replication, safe data mode, that timed out. This attribute is meaningless if the remote server is a directory server.

7.8 Overview of Fractional Replication

The fractional replication feature enables you to restrict certain attributes from being included when modify operations are replayed on specific servers in a topology.

For information about configuring fractional replication, see Configuring Fractional Replication.

This section describes the architecture of the fractional replication mechanism and covers the following topics:

- About Fractional Data Set Identification
- About Fractional Replication Filtering
- About Fractional Replication and Local Operations

7.8.1 About Fractional Data Set Identification

A fractional data set is identified by the operational attributes that are stored in the root entry of the replicated domain.

The following operational attributes help you to identify a fractional data set:

- `ds-sync-fractional-exclude`
- `ds-sync-fractional-include`

The syntax and meaning of these attributes is identical to their corresponding configuration attributes, described in Configuring Fractional Replication. The role of these operational attributes is to *tag* a data set as fractional: their presence in a domain implies "this data set is a fractional domain and does not contain the following specific attributes...".

The fractional configuration stored in the root entry of the domain, combined with the generation ID (`ds-sync-generation-id`) and the replication state (`ds-sync-state`), can be seen as the *fractional signature* of the data set.

When a domain is enabled (for example, after its fractional configuration is modified), the server compares the fractional configuration of the domain (under `cn=config`) with the fractional configuration attributes in the root entry of the domain. If both configurations match, the domain assumes a *normal* status and LDAP operations can be accepted. If the configurations do not match, the domain assumes a *bad generation ID* status and the data set must be synchronized (by importing a data set) before LDAP operations can be accepted.

The data set that is imported must either:

- have the same fractional configuration in its root entry as the local domain has under `cn=config`. In this case, the data set is imported as is.
- have no fractional configuration in its root entry. In this case, the data set is imported and filtered according to the attribute filtering rules defined in the fractional configuration of the local domain (under `cn=config`). The `ds-sync-fractional-exclude` or `ds-sync-fractional-include` attributes are then created in the root entry of the imported data, by copying the fractional configuration of the local domain.

7.8.2 About Fractional Replication Filtering

You can filter the operations that have to be replayed while configuring a domain as fractional. This is referred to as fractional replication filtering.

When a domain is configured as fractional, all `ADD`, `MODIFY`, and `MODIFYDN` operations that arrive from the network to be replayed are filtered. These operations can end up being abandoned if all of the attributes in the operation are filtered attributes according to the fractional configuration.

7.8.3 About Fractional Replication and Local Operations

If an LDAP client performs an operation directly on a fractional replica and the operation does not match the fractional configuration, the operation is forbidden and the server returns an "unwilling to perform" error.

For example, if a fractional replica is configured with `fractional-exclude: *:jpegPhoto` and an LDAP client attempts to add a new entry that contains a `jpegPhoto` attribute, the operation is rejected with an "unwilling to perform" error. This behavior ensures that the domain remains consistent with its fractional configuration definition, which implies that no `jpegPhoto` attribute can exist on the domain.

8

Understanding the Oracle Unified Directory Indexing Model

Oracle Unified Directory supports multiple kinds of indexes. You must explore this section to learn more about the index types and the manner is which the search is conducted.
This section contains the following topics:

- Overview of Indexes
- Supported Index Types
- What is Index Entry Limit?
- Understanding the Search Evaluation Mechanism
- Maintaining Indexes

For information about configuring indexes, see Indexing Directory Data.

8.1 Overview of Indexes

Oracle Unified Directory uses indexes to speed up search operations by associating lookup information with Oracle Unified Directory entries. Each search operation includes a search key that specifies the entries to return. During a search operation the server uses the index to find entries that match the search key. If indexes are not configured, then the server must check every entry in a suffix to locate potential matches for the search key.

Navigating through all entries in the directory is resource-intensive, especially for large directories. In addition, unindexed searches might not be allowed to non-privileged users. For more information about assigning privilege for unindexed search, see Understanding Root Users and the Privilege Subsystem. To make searches more efficient, you can configure indexes to correspond to the searches that clients need to perform.

This section contains the following topics:

- About Indexes
- Understanding the Importance of Indexing

8.1.1 About Indexes

An index is a mechanism used by the Directory Server database to efficiently find entries matching the search criteria. An index maps a search key to an ID list, which is a set of entry IDs for the entries that match that index key.

When you perform a search operation, Directory Server uses the index to find entries that match the index key. Without an index, Directory Server must check every entry in a suffix to find the match.

8.1.2 Understanding the Importance of Indexing

Indexes play an important role in enhancing the performance of search operations. Directory Server indexes speed up searches by associating search strings with the contents of a directory.

- The most efficient methodology to improve search operations against the directory server is to configure indexes, combined with defining an index entry limit on search results.
- An index stores the values of specified attributes for an entry without storing any other detail about the entry. This saves space and makes search faster by organizing the index around that attribute. If you perform a search on an attribute that has been indexed, Oracle Unified Directory quickly locates the index for the entries that meet the search criteria.

8.2 Supported Index Types

An index is an optional structure to speed up data access. Oracle Unified Directory supports various index types.

The following table lists the types of indexes:

Table 8-1 Supported index types

Type of index	Description
Approximate Indexes	An approximate index is used to match values that sound like the values that are provided in the search filter. The purpose of an approximate index is to locate entries that match values similar to the search filter. For example, an approximate index on the `cn` attribute allows client applications to locate entries even when the names are misspelled.
Equality Indexes	An equality index identifies which entries are exactly equal to the value that is provided in a search filter. An equality index can only be maintained for attributes that have a corresponding equality matching rule.
Ordering Indexes	An ordering index keeps track of the relative order of values for an attribute. It is similar to an equality index, except that it uses an ordering matching rule instead of an equality matching rule to normalize the values. Ordering indexes cannot be maintained for attributes that do not have a corresponding ordering matching rule.
Presense Indexes	A presence index keeps track of the entries that have at least one value for a specified attribute. There is only a single presence index key per attribute, and its ID list contains the entry ID for all entries that contain the specified attribute. The `aci` attribute is indexed for presence by default to enable quick retrieval of entries with ACIs.
Substring Indexes	A substring index keeps track of which entries contain specific substrings. Index keys for a substring index consist of six-character substrings taken from attribute values and the corresponding values are an ID list containing the entry ID of the entries containing those substrings. The attribute's substring matching rule is used to normalized value the values for the index keys, and substring indexes cannot be defined for attributes that do not contain substring matching rules.

8.3 What is Index Entry Limit?

The index entry limit is a configuration limit.

The index entry limit can be used to control the maximum number of entries that is allowed to match any given index key (that is, the maximum size of an ID list). This provides a mechanism for limiting the performance impact for maintaining index keys that match a large percentage of the entries in the server. In cases where large ID lists might be required, performing an unindexed search can often be faster than one that is indexed.

8.4 Understanding the Search Evaluation Mechanism

To process an LDAP search operation, the server applies each assertion of the search filter to generate a list of candidate entries, which are then combined to form an initial set of candidate entry IDs.

If a candidate set is obtained, the search is considered to be **indexed.** Each candidate entry is fetched from the entry database and returned to the client if it matches the search scope and filter.

If no candidate set is obtained (because of a lack of indexes or because some index values exceeded the index entry limit), the search is considered to be **not-indexed.** In this case, a cursor is opened on the DN database at the base entry to iterate through all records in scope, fetching and filtering the corresponding entries until all the entries under the search base have been processed.

Whenever the number of candidate entry IDs from the indexes is found to be 10 or less, no further attempt is made to reduce the number of candidates. Instead those entries are immediately fetched from the entry database and filtered, on the assumption that this is quicker than continuing to read the index databases. This can pay off for AND search filters in which the first component is the most specific.

Search AND filters are also rearranged so that components that are slow to evaluate (greater-than-or-equal, less-than-or-equal) come after components that are generally faster (for example, equality).

8.5 Maintaining Indexes

You can maintain indexes by running some commands.

Consider the following key points for maintaining indexes:

- Run the `verify-index` command to check the consistency between the index and the entry data within the directory server database.

  For more information about the command, see verify-index
- Run the `rebuild-index` command to rebuild the directory server indexes, if you create a new index or when the `index-entry-limit` property of an index changes.

  For more information about the command, see rebuild-index
- Configure a Virtual List View (VLV) index, which is a mechanism used by the Directory Server database to efficiently process searches with VLV controls. A VLV index effectively notifies the server that a virtual list view, with specific query and sort parameters, will be performed. This index also allows the server to collect and maintain the information

required to make using the virtual list view faster. A VLV index stores sorted blocks of ID lists, which are a set of entry IDs and the attribute values of the entry to sort on.

For more information about configuring VLV indexes, see Configuring VLV Indexes.

- Configure an extensible match index to accelerate search operations using an extensible match search filter. Index keys are values that have been normalized using a specified matching rule, and the corresponding ID list contains the entry ID for all entries that match the value according to that matching rule.

  For more information about extensible match search filter, see extensible match search filter.

9

Understanding Access Control Model in Oracle Unified Directory

Access control is a mechanism that is used to regulate access to resources in a computing environment.
The following topics provide descriptive and reference information about the directory server access control model:

- Understanding Access Control Principles
- Understanding the Syntax of Access Control Instructions
- Understanding Bind Rules
- Understanding Bind Rule Syntax
- Compatibility With the Oracle Directory Server Enterprise Edition Access Control Model
- Using Macro ACIs for Advanced Access Control
- Understanding Virtual Access Control Instructions

For information about configuring access control in the directory server, see Controlling Access To Data.

9.1 Understanding Access Control Principles

You need to understand the principles of the access control mechanism provided with the directory server to configure access control policies.

This section contains the following topics:

- About Access Control
- Overview of Access Control Instructions Structure
- Configuring Directory Server Global Access Control Instructions
- About Evaluation of Access Control Instructions
- About Limitations of Access Control Instructions
- About Replication of Access Control Instructions
- About Anonymous Read Access ACI

See also Managing Global ACIs Using `dsconfig`.

9.1.1 About Access Control

When the directory server receives a request, it uses the authentication information provided by the user in the bind operation, and the access control instructions (ACIs) defined in the server to allow or deny access to directory information.

The server can allow or deny permissions such as read, write, search, or compare. The permission level granted to a user might depend on the authentication information that the user provides.

Using access control, you can control access to the entire directory, a subtree of the directory, specific entries in the directory (including entries that define configuration tasks), a specific set of entry attributes, or specific entry attribute values. You can set permissions for a particular user, for all users who belong to a specific group or role, or for all users of the directory. Finally, you can define access for a specific client, identified by its IP address or DNS name.

9.1.2 Overview of Access Control Instructions Structure

ACIs are used to allow or deny access to directory information. ACIs are stored in the directory as attributes of entries.

The `aci` attribute is an operational attribute that is available for use on every entry in the directory, regardless of whether it is defined for the object class of the entry. This attribute is used by the directory server to evaluate what rights are granted or denied when the directory server receives an LDAP request from a client. The `aci` attribute is returned in an `ldapsearch` operation only if it is specifically requested.

An ACI statement includes three main parts:

Target
Determines the entry or attributes to which permissions apply.

Permission
Defines what operations are allowed or denied.

Bind Rule
Determines who is subject to the ACI, based on their bind DN.

The permission and bind rule portions of the ACI are set as a pair, also called an Access Control Rule (ACR). The specified permission to access the target is granted or denied depending on whether the accompanying rule is evaluated to be true. For more information, see Understanding the Syntax of Access Control Instructions.

If an entry that contains an ACI does not have child entries, the ACI applies to that entry only. If the entry has child entries, the ACI applies to the entry itself and to all entries below it. Therefore, when the directory server evaluates access permissions to an entry, it verifies the ACIs for every entry between the one that was requested and the base of its root suffix.

The `aci` attribute is multivalued, which means that you can define several ACIs for the same entry or subtree.

You can create an ACI on an entry that does not apply directly to that entry but to some or all of the entries in the subtree below it. The advantage of this is that you can place at a high level in the directory tree a general ACI that effectively applies to entries that are more likely to be located lower in the tree. For example, at the level of an `organizationalUnit` entry or a `locality` entry, you could create an ACI that targets entries that include the `inetorgperson` object class.

You can use this feature to minimize the number of ACIs in the directory tree by placing general rules at high-level branch points. To limit the scope of more specific rules, place them as close as possible to leaf entries.

> **Note:**
>
> ACIs that are placed in the root DSE entry (with the DN `""`) apply only to that entry.

9.1.3 Configuring Directory Server Global Access Control Instructions

You can configure access control centrally by using the `dsconfig` command to modify the properties of the Access Control Handler.

The following default global ACIs apply to all suffixes that are defined in the directory server because the rules do not specify a `target` expression:

```
Property   : Value(s)
-----------:---------------------------------------------------------------
global-aci : "(targetattr="*")(version 3.0; acl "Self entry modification";
           : allow (write) userdn="ldap:///self";),
           : "(targetattr="createTimestamp||creatorsName||modifiersName||modify
           : Timestamp||entryDN||entryUUID||subschemaSubentry")(version 3.0;
           : acl "User-Visible Operational Attributes"; allow
           : (read,search,compare) userdn="ldap:///anyone";)",
```

For more information, see Managing Global ACIs Using `dsconfig`.

9.1.4 About Evaluation of Access Control Instructions

To evaluate the access rights to a particular entry, the server compiles a list of the ACIs present on the entry itself and on the parent entries back up to the base of the entry's root suffix. During evaluation, the server processes the ACIs in this order.

ACIs are evaluated in all of the suffixes and subsuffixes between an entry and the base of its root suffix, but not across chained suffixes on other servers.

> **Note:**
>
> Access control does not apply to any user who has the `bypass-acl` privilege. The Directory Manager has this privilege. When a client is bound to the directory as the Directory Manager, the directory server does not evaluate any ACIs before performing operations. As a result, performance of LDAP operations as Directory Manager is not comparable to the expected performance of other users. You should always test directory performance with a typical user identity.

By default, if no ACI applies to an entry, access is denied to all users except those with the `bypass-acl` privilege. Access must be explicitly granted by an ACI for a user to access any entry in the directory. For more information, see About Default Global ACIs.

Although the directory server processes the ACIs that are closest to the target entry first, the effect of all ACIs that apply to an entry is cumulative. Access granted by any ACI is allowed unless any other ACI denies it. ACIs that deny access, no matter where they appear in the list, take precedence over ACIs that allow access to the same resource.

For example, if you deny write permission at the directory's root level, none of the users can write to the directory regardless of the specific permissions you grant them. To grant a specific user write permissions to the directory, you must restrict the scope of the original denial for write permission so that it does not include that user.

9.1.5 About Limitations of Access Control Instructions

There are some limitations that you must bear in mind when you create an access control policy for your directory service.

The limitations are as follows:

- If your directory tree is distributed over several directory servers, some restrictions apply to the keywords that you can use in access control statements. ACIs that depend on group entries (`groupdn` keyword) must be located on the same directory server as the group entry. If the group is dynamic, all members of that group must also have an entry on the directory server. If the group is static, the members' entries can be located on remote directory servers. However, you can do value matching of values stored in the target entry with values stored in the entry of the bind user (for example, using the `userattr` keyword). Access is evaluated normally even if the bind user does not have an entry on the directory server that holds the ACI.
- Access control rules are always evaluated on the local directory server. You must not specify the host name or port number of the directory server in LDAP URLs used in ACI keywords. If you do, the LDAP URL is not taken into account at all.

9.1.6 About Replication of Access Control Instructions

ACIs are stored as attributes of entries, so if an entry containing ACIs is part of a replicated suffix, the ACIs are replicated like any other attribute.

9.1.7 About Anonymous Read Access ACI

Anonymous read access ACI is automatically added to a server instance during the Oracle Unified Directory setup when you enable an instance as a datastore for Oracle Enterprise User Security (EUS).

```
"(targetattr!="userPassword||authPassword")(version 3.0; acl "Anonymous read access";
allow (read,search,compare) userdn="ldap:///anyone";)"
```

To enable a server instance as a datastore for EUS, you must select the **Enable for EUS (Enterprise User Security), EBS, Database Net Services and DIP** option in the **Oracle Components Integration** window during the Oracle Unified Directory setup. See Setting Up Oracle Unified Directory as a Directory Server in *Installing Oracle Unified Directory*.

9.2 Understanding the Syntax of Access Control Instructions

ACIs are complex structures with many possible variations. A detailed analysis of the ACI syntax is dealt with in this section.

This section contains the following topics:

- Overview of Access Control Instructions Syntax
- Defining Targets
- Setting Permissions

See also Understanding Bind Rule Syntax.

9.2.1 Overview of Access Control Instructions Syntax

You must review the ACI syntax to regulate access to directory data.

The `aci` attribute has the following syntax:

```
aci: (target)(version 3.0;acl "name";permissionBindRules;)
```

where:

- *target* specifies the entry, attributes, or set of entries and attributes for which you want to control access. The target can be a distinguished name, one or more attributes, or a single LDAP filter. The target is optional. When the target is not specified, the ACI applies to the entire entry where it is defined and all of its children.
- `version 3.0` is a required string that identifies the ACI version.
- *name* is a name for the ACI. The name can be any string that identifies the ACI. The ACI name is required and should describe the effect of the ACI. Although there are no restrictions on the name, it is good practice to use unique names for ACIs. If you use unique names, the Get Effective Rights control enables you to determine which ACI is in force.
- *permission* specifically states what rights you are either allowing or denying, for example read or search rights.
- *bindRules* specify the credentials and bind parameters that a user has to provide to be granted access. Bind rules can also be based on user or group membership or connection properties of the client.

You can specify multiple targets and permission-bind rule pairs. This allows you to refine both the entry and attributes being targeted and efficiently set multiple access controls for a given target, as shown here:

```
aci: (target)...(target)(version 3.0;acl "name"; permissionBindRule;
permissionBindRule; ...; permissionBindRule;)
```

The following example shows a complete LDIF ACI:

```
aci: (target="ldap:///uid=bjensen,dc=example,dc=com")
(targetattr="*")(version 3.0; acl "example"; allow (write)
userdn="ldap:///self";)
```

In this example, the ACI states that the user `bjensen` has rights to modify all attributes in her own directory entry.

9.2.2 Defining Targets

The target identifies what the ACI applies to. When a client requests an operation on attributes in an entry, the directory server evaluates the target to see if the ACI must be evaluated to allow or deny the operation.

If the target is not specified, the ACI applies to all attributes in the entry containing the `aci` attribute and to the entries below it.

The following sections describe how to define targets:

- Overview of LDIF Target Keywords
- Targeting a Directory Entry

- Targeting Attributes in a Targeted Entry
- Targeting Both an Entry and Attributes
- Targeting Entries or Attributes Using LDAP Filters
- Targeting Attribute Values Using LDAP Filters
- Targeting a Single Directory Entry
- Specifying the Scope of an ACI
- Targeting LDAP Controls
- Targeting LDAP Extended Operations

9.2.2.1 Overview of LDIF Target Keywords

The general syntax for a target is one of the following:

```
(keyword = "expression")
(keyword != "expression")
```

where:

- *keyword* indicates the type of target. The following types of targets are defined by the keywords in Table 9-1:
  - A directory entry or its subtree
  - The attributes of an entry
  - A set of entries or attributes that match an LDAP filter
  - An attribute value or combination of values that match an LDAP filter
  - The scope of the ACI
  - An LDAP control
  - An extended operation
- The equal sign (=) indicates that the target is the object specified in the expression, and not equal (!=) indicates that the target is any object not specified in the expression.

> **Note:**
>
> The not-equal operator is not supported for the `targattrfilters` and `targetscope` keywords.

- *expression* depends on the keyword and identifies the target. The quotation marks ("") around *expression* are syntactically required, although the current implementation accepts expressions like `targetattr=*`. In future versions, syntax checking might become more strict, so you should always use quotation marks.

The following table lists each keyword and the associated expressions.

Table 9-1 LDIF Target Keywords

Keyword	Valid Expressions	Wildcard Allowed?
`target`	`ldap:///`*distinguishedName*	Allowed

Table 9-1 (Cont.) LDIF Target Keywords

Keyword	Valid Expressions	Wildcard Allowed?
`targetattr`	*attribute*	Allowed
`targetfilter`	*LDAPfilter*	Allowed
`targattrfilters`	*LDAPoperation*:*LDAPfilter*	Allowed
`targetscope`	`base, onelevel, subtree, subordinate`	Not Allowed
`targetcontrol`	*oid*	Not Allowed
`extop`	*oid*	Not Allowed

9.2.2.2 Targeting a Directory Entry

Use the target keyword and a DN inside an LDAP URL to target a specific directory entry and any entries below it. The targeted DN must be located in the entry where the ACI is defined or in the subtree below the entry. The target expression has the following syntax:

```
(target = "ldap:///distinguishedName")
(target != "ldap:///distinguishedName")
```

The distinguished name must be located in the entry where the ACI is defined or in the subtree below the entry. For example, the following target can be used in an ACI on `ou=People,dc=example,dc=com`:

```
(target = "ldap:///uid=bjensen,ou=People,dc=example,dc=com")
```

The keyword `target` is optional. If it is not present, the default target for the ACI is the entry where the ACI is stored.

> **Note:**
>
> The DN of the entry must be a distinguished name in string representation (as defined in RFC 4514 (http://www.ietf.org/rfc/rfc4514.txt)). Therefore, characters syntactically significant for a DN, such as commas, must be escaped with a single backslash (\). For example:
>
> ```
> (target="ldap:///uid=cfuentes,o=Example Bolivia\, S.A.")
> ```

You can also use a wildcard in the DN to target any number of entries that match the LDAP URL. The following are legal examples of wildcard usage:

- `(target="ldap:///uid=*,dc=example,dc=com")` Matches every immediate child of the `example.com` branch entry that has the `uid` attribute in the entry's RDN, as shown in this example.

  ```
  uid=tmorris,dc=example,dc=com
  uid=yyorgens,dc=example,dc=com
  uid=bjensen,dc=example,dc=com
  ```

- `(target="ldap:///uid=*,**,dc=example,dc=com")` Matches every entry more than one level below the `example.com` branch entry that has the `uid` attribute in the entry's RDN, as shown in this example.

  ```
  uid=tmorris,ou=sales,dc=example,dc=com
  uid=yyorgens,ou=marketing,dc=example,dc=com
  uid=bjensen,ou=eng,ou=east,dc=example,dc=com
  ```

- `(target="ldap:///uid=*Anderson,ou=People,dc=example,dc=com")` Matches every entry immediately below the `ou=People` branch entry with a `uid` ending in Anderson.
- `(target="ldap:///*=*Anderson,ou=People,dc=example,dc=com")` Matches every entry immediately below the `ou=People` branch whose RDN ends with Anderson, regardless of the naming attribute.

Multiple wildcards are allowed, such as in `uid=*,ou=*,dc=example,dc=com`, which matches every entry in the `example.com` tree whose distinguished name contains the `uid` and `ou` attributes in the specified positions.

9.2.2.3 Targeting Attributes in a Targeted Entry

In addition to targeting directory entries, you can also target one or more attributes that occur in the targeted entries. This functionality is useful when you want to deny or allow access to partial information about an entry. For example, you can allow access to only the common name, surname, and telephone number attributes of a given entry. Similarly, you can deny access to sensitive information such as personal data.

If no `targetattr` rule is present, no attributes can be accessed by default. To access all attributes, the rule must be `targetattr="*"`.

The targeted attributes do not need to exist on the target entry or its subtree, but the ACI applies whenever they do. The attributes you target do not need to be defined in the schema. The absence of schema checking makes it possible to implement an access control policy before importing your data and its schema.

To target attributes, use the `targetattr` keyword and provide the attribute names. The `targetattr` keyword uses the following syntax:

```
(targetattr = "attribute")
(targetattr != "attribute")
```

You can target multiple attributes by using the `targetattr` keyword with the following syntax:

```
(targetattr = "attribute1 || attribute2 ... || attributeN")
(targetattr != "attribute1 || attribute2 ... || attributeN")
```

For example, to target an entry's common name, surname, and UID attributes, you would use the following:

```
(targetattr = "cn || sn || uid")
```

To target all of an entry's user attributes, except `carlicense`, you would use the following target:

```
(targetattr != "carlicense")
```

Targeted attributes include all subtypes of the named attribute. For example, `(targetattr = "locality")` also targets `locality;lang-fr`. You can also target subtypes specifically, for example, `(targetattr = "locality;lang-fr-ca")`.

You can use a wildcard as a stand-alone character in a `targetattr` rule (such as `targetattr="*"`), but this use is discouraged because it serves no particular purpose and can have a negative performance impact.

9.2.2.4 Targeting Both an Entry and Attributes

By default, the entry targeted by an ACI containing a `targetattr` keyword is the entry on which the ACI is placed. That is, if you apply the ACI `aci: (`*`targetattr`* `= "`*`uid`*`") (`*`accessControlRules`*`;)` to the `ou=Marketing, dc=example,dc=com` entry, then the ACI applies to the entire Marketing subtree. However, you can also explicitly specify a target using the target keyword, as shown in the following example:

```
aci: (target="ldap:///uid=*,ou=Marketing,dc=example,dc=com")
(targetattr="uid") (accessControlRules;)
```

The order in which you specify the target and the `targetattr` keywords is irrelevant.

9.2.2.5 Targeting Entries or Attributes Using LDAP Filters

Use LDAP filters to target a set of entries that match certain criteria. To do this, use the `targetfilter` keyword with an LDAP filter. The ACI applies to all entries that match the filter at the level of the target DN and in the subtree below it.

The `targetfilter` keyword uses this syntax:

```
(targetfilter = "LDAPfilter")
```

where *LDAPfilter* is a standard LDAP search filter. For more information about filter syntax, see search filter.

For example, suppose that all entries representing employees have a status of salaried or contractor and an attribute representing the number of hours worked, as a percentage of a full-time position. To target all the entries representing contractors or part-time employees, you could use the following filter:

```
(targetfilter = "(|(status=contractor)(fulltime<=79))")
```

The Netscape extended filter syntax is not supported in ACIs. For example, the following target filter is not valid:

```
(targetfilter = "(locality:fr:=<= Quebec)")
```

Target filters select whole entries as targets of the ACI. You can associate the `targetfilter` and the `targetattr` keywords to create ACIs that apply to a subset of attributes in the targeted entries.

The following LDIF example allows members of the Engineering Admin group to modify the `departmentNumber` and `manager` attributes of all entries in the Engineering business category. This example uses LDAP filtering to select all entries with `businessCategory` attributes set to `Engineering`:

```
dn: dc=example,dc=com
objectClass: top
objectClass: organization
aci: (targetattr="departmentNumber || manager")
(targetfilter="(businessCategory=Engineering)")
(version 3.0; acl "eng-admins-write"; allow (write)
groupdn ="ldap:///cn=Engineering Admins, dc=example,dc=com";)
```

Although using LDAP filters can be useful when you are targeting entries and attributes that are spread across the directory, the results are sometimes unpredictable because filters do not directly name the object for which you are managing access. The set of entries targeted by a filtered ACI is likely to change as attributes are added or deleted. Therefore, if you use LDAP filters in ACIs, you should verify that they target the correct entries and attributes by using the same filter in an `ldapsearch` operation.

9.2.2.6 Targeting Attribute Values Using LDAP Filters

Use access control to target specific attribute values. This means that you can grant or deny permissions on an attribute if that attribute's value meets the criteria defined in the ACI. An ACI that grants or denies access based on an attribute's value is called a value-based ACI.

For example, you can grant all users in your organization permission to modify the `roomNumber` attribute in their own entries. However, you would also want to ensure that they do not give themselves reserved room numbers, all of which begin with `12`. LDAP filters are used to check that the conditions on attribute values are satisfied.

To create a value-based ACI, you must use the `targattrfilters` keyword with the following syntax:

`(targattrfilters="`*Op*`=`*attr1*`:`*F1*`[(&&` *attr2*`:`*F2*`)*][;`*Op*`=`*attr*`:`*F*`[(&&` *attr*`:`*F*`)*]")`

where:

- *Op* is either an `add` or `delete` operation:
  - `add` represents the operation of creating an attribute.
  - `delete` represents the operation of deleting an attribute.
- *attr* represents the target attributes.
- *F* represents search filter that applies only to the associated attribute.

When creating an entry, if a filter applies to an attribute in the new entry, then all values of that attribute must satisfy the filter. When deleting an entry, if a filter applies to an attribute in the entry, then all values of that attribute must also satisfy the filter.

When modifying an entry, if the operation adds an attribute, then the add filter that applies to that attribute must be satisfied. If the operation deletes an attribute, then the delete filter that applies to that attribute must be satisfied. If individual values of an attribute already present in the entry are replaced, then both the add and delete filters must be satisfied.

The following example attribute filter allows users to add any `roomNumber` attribute to their own entries except the reserved room numbers, which have a `12` prefix. It also allows users to add a telephone number with a `123` prefix.

```
(targattrfilters="add=roomNumber:(!(roomNumber=12*)) && telephoneNumber:
(telephoneNumber=123*)")
```

9.2.2.7 Targeting a Single Directory Entry

There is no explicit way to target a single entry. However, you can achieve this in one of two ways:

- By creating a bind rule that matches user input in the bind request with an attribute value stored in the targeted entry
- By using the `targetfilter` keyword

With the `targetfilter` keyword you can specify an attribute value that appears only in the desired entry. For example, during the installation of the directory server, the following ACI is created:

```
aci: (targetattr="*")(targetfilter=(o=example))
(version 3.0; acl "Default anonymous access";
allow (read, search) userdn="ldap:///anyone";)
```

This ACI can apply only to the `o=example` entry, because that is the only entry with an attribute `o` having the value `example`.

The risk associated with these methods is that your directory tree can change in the future, and you would have to remember to modify this ACI.

9.2.2.8 Specifying the Scope of an ACI

Usually an ACI has subtree scope. You can restrict the scope of an ACI by using the `targetscope` keyword with the following syntax:

`(targetscope="`*expression*`")`

where *expression* is one of the following:

`base`
The ACI applies to the target resource only.

`onelevel`
The ACI applies to the target resource's first-generation children.

`subtree`
The ACI applies to the target resource and the subtree below it.

`subordinate`
The ACI applies only to the subtree below the target resource.

If the `targetscope` is not specified, the default value is `subtree`. The following example restricts the ACI target match only to the entry with the distinguished name `uid=bjensen,ou=People,dc=example,dc=com` and any of the children one level below it:

```
(target = "ldap:///uid=bjensen,ou=People,dc=example,dc=com")(targetscope="onelevel")
```

> **Note:**
>
> The not-equal operator is not supported for the `targetscope` keyword.

9.2.2.9 Targeting LDAP Controls

To target LDAP controls, use the `targetcontrol` keyword and provide the control object identifier. The `targetcontrol` keyword uses the following syntax:

`(targetcontrol="`*oid*`")`

`(targetcontrol!="`*oid*`")`

You can target multiple LDAP controls by using the `targetcontrol` keyword with the following syntax:

```
(targetcontrol="oid1 || oid2... || oidN")
```

```
(targetcontrol!="oid1 || oid2... || oidN")
```

For example, to target both the get effective rights control and the proxied authorization control, use the following `targetcontrol` expression:

```
(targetcontrol = "1.3.6.1.4.1.42.2.27.9.5.2 || 2.16.840.1.113730.3.4.18")
```

> **Note:**
>
> The get effective rights control has OID value of 1.3.6.1.4.1.42.2.27.9.5.2. The proxy authorization V2 control has OID value of 2.16.840.1.113730.3.4.18.

9.2.2.10 Targeting LDAP Extended Operations

To target extended operations, use the `extop` keyword and provide the operation object identifier. The `extop` keyword uses the following syntax:

```
(extop= "oid")
```

```
(extop!= "oid")
```

You can target multiple extended operations by using the `extop` keyword with the following syntax:

```
(extop = "oid1 || oid2... || oidN")
```

```
(extop!= "oid1 || oid2... || oidN")
```

For example, to target both the StartTLS extended operation and the Password Modify extended operation, use the following `extop` expression:

```
(extop = "1.3.6.1.4.1.1466.20037 || 1.3.6.1.4.1.4203.1.11.1.")
```

> **Note:**
>
> Access control using the `extop` keyword with a StartTLS extended operation target must always be done using Global ACIs. The authorization entry in the StartTLS extended operation is null.

9.2.3 Setting Permissions

Permissions specify the type of access that you are allowing or denying. You can either allow or deny permission to perform specific operations in the directory. The various operations that can be assigned are known as rights.

There are two parts to setting permissions:

- Allowing or denying access
- Assigning rights

The following sections describe how to define permissions:

9.2.3.1 About Access Permissions

You can explicitly allow or deny access permissions by using the `allow` or the `deny` keyword.

9.2.3.2 Overview of Rights Assignment

Rights detail the specific operations a user can perform on directory data. You can allow or deny all rights, or you can assign one or more of the following rights:

Read
Indicates whether users can read the directory entries and the attributes of entries specified in the ACI. This permission applies only to the search operation. (Compare the Read permission with the description of the Search permission that follows.)

Write
Indicates whether users can modify an entry by adding, modifying, or deleting attributes. This permission applies to the modify and modRDN operations.

Add
Indicates whether users can create entries. This permission applies only to the add operation.

Delete
Indicates whether users can delete entries. This permission applies only to the delete operation.

Search
Indicates whether users can search on the targets specified in the ACI. This permission applies only to the search operation. The Search right is checked once, and after the search is allowed or denied, it is not checked again. If the search is allowed, the read right is then applied to each entry to be returned as a result of the search and to each attribute of each entry.

Compare
Indicates whether users can compare data they supply with data stored in the directory. With compare rights, the directory returns a success or failure message in response to an inquiry, but the user cannot see the value of the entry or attribute. This permission applies only to the compare operation.

Selfwrite
Indicates whether users can add or delete their own DN in an attribute of the target entry. The syntax of this attribute must be a distinguished name. This right is used only for group management. Selfwrite works with proxy authorization: it grants the right to add or delete the proxy DN from the group entry (not the DN of the bound user).

Proxy
Indicates whether the specified DN can access the target with the rights of another entry. You can grant proxy access using the DN of any user in the directory except the Directory Manager DN. Moreover, you cannot grant proxy rights to the Directory Manager. An example is provided in About Proxy Authorization ACIs.

Import
Used by the modify DN operation. This access right indicates whether an entry can be imported to the specified DN.

Export
Used by the modify DN operation. This access right indicates whether an entry can be exported from the specified DN.

All
Indicates that the specified DN has the following rights to the targeted entry: read, write, search, delete, compare, and selfwrite. The All access right does not give the following rights to the target entry: proxy, import, and export.

Rights are granted independently of one another. This means, for example, that a user who is granted add rights can create an entry but cannot delete it if delete rights have not been specifically granted. Therefore, when planning the access control policy for your directory, you must ensure that you grant rights in a way that makes sense for users. For example, it does not usually make sense to grant write permission without granting read and search permissions.

9.2.3.3 Overview of Rights Required for LDAP Operations

This section describes the rights that you must grant to users depending on the type of LDAP operation that you want to authorize them to perform.

- Adding an entry
  - Grant add permission on the entry being added.
  - Grant write permission on the value of each attribute in the entry. This right is granted by default but could be restricted using the `targattrfilters` keyword.
- Deleting an entry
  - Grant delete permission on the entry to be deleted.
  - Grant write permission on the value of each attribute in the entry. This right is granted by default but could be restricted using the `targattrfilters` keyword.
- Modifying an attribute in an entry
  - Grant write permission on the attribute type.
  - Grant write permission on the value of each attribute type. This right is granted by default but could be restricted using the `targattrfilters` keyword.
- Modifying the RDN of an entry
  - Grant write permission on the entry.
  - Grant write permission on the attribute type used in the new RDN.
  - Grant write permission on the attribute type used in the old RDN, if you want to grant the right to delete the old RDN.
  - Grant write permission on the value of the attribute type used in the new RDN. This right is granted by default but could be restricted using the `targattrfilters` keyword.
- Moving an entry to another subtree
  - Grant export permissions on the entry that you want to move.
  - Grant import permission on the new superior entry of the entry that you want to move.
- Comparing the value of an attribute

  - Grant compare permission on the attribute type.
- Searching for entries
  - Grant search permission on each attribute type used in the search filter.
  - Grant read permission on at least one attribute type used in the entry to ensure that the entry is returned.
  - Grant read permission an each attribute type to be returned with the entry.

The following example better illustrates the permissions that you must configure to enable users to search the directory. Consider the following search:

```
$ ldapsearch -h host -p port -D "uid=bjensen,dc=example,dc=com" \
  -j pwd-file -b "dc=example,dc=com" \
  "(objectclass=*)" mail
```

The following ACI is used to determine whether user `bjensen` can be granted access for searching her own entry:

```
aci: (targetattr = "mail")(version 3.0; acl "self access to \
mail"; allow (read, search) userdn = "ldap:///self";)
```

The search result list is empty because this ACI does not allow `bjensen` the right to search on the `objectclass` attribute. For the search operation to be successful, you must modify the ACI, as shown in the following example:

```
aci: (targetattr = "mail || objectclass")(version 3.0; acl \
"self access to mail"; allow (read, search) userdn = \
"ldap:///self";)
```

9.2.3.4 About Permissions in ACI Statement

In an ACI statement, permissions use the following syntax:

`allow|deny (`*rights*`)`

where *rights* is a list of comma-separated keywords enclosed within parentheses. Valid keywords are `read`, `write`, `add`, `delete`, `search`, `compare`, `selfwrite`, `proxy`, `import`, `export`, or `all`.

The `all` access right does not give the following rights to the target entry: `proxy`, `import`, and `export`.

In the following example, `read`, `search`, and `compare` access is allowed, if the bind rule is evaluated to be true:

```
aci: (target="ldap:///dc=example,dc=com") (version 3.0;acl \
"example"; allow (read, search, compare) bindRule;)
```

9.3 Understanding Bind Rules

Depending on the ACIs defined for the directory, for certain operations, you must bind to the directory. You must analyze the bind rules to control access to directory information.

The following sections describe how bind rules are used to control access:

- Overview of Bind Rules
- Using Boolean Bind Rules

9.3.1 Overview of Bind Rules

Binding means logging in or authenticating yourself to the directory by providing a bind DN and password, or, if using SSL, a certificate. The credentials provided in the bind operation and the circumstances of the bind determine whether access to the directory is allowed or denied.

Every permission set in an ACI has a corresponding bind rule that details the required credentials and bind parameters.

A simple bind rule might require that the person accessing the directory belong to a specific group. A complex bind rule can state that a person must belong to a specific group and must log in from a machine with a specific IP address between 8 a.m. and 5 p.m.

Bind rules define who can access the directory, when, and from where. More specifically, bind rules can specify the following:

- Users, groups, and roles that are granted access
- Location from which an entity must bind (The location from which a user authenticates can be spoofed and can therefore not be trusted. Do not base ACIs on this information alone.)
- Time or day on which binding must occur
- Type of authentication that must be in use during binding
- Security strength factor (that is, the length of encryption key currently in use)

Additionally, bind rules can be complex constructions that combine these criteria by using Boolean operators, as described in Understanding Bind Rule Syntax.

The directory server evaluates the logical expressions used in ACIs according to a three-valued logic similar to the one used to evaluate LDAP filters, as described in RFC 4511 (`http://www.ietf.org/rfc/rfc4511.txt`) Lightweight Directory Access Protocol (LDAP): The Protocol. In summary, this means that if any component in the expression evaluates to Undefined (for example if the evaluation of the expression aborted due to a resource limitation), then the directory server handles this case correctly: it does not erroneously grant access because an Undefined value occurred in a complex Boolean expression.

9.3.2 Using Boolean Bind Rules

Bind rules can be complex expressions that use the Boolean expressions `AND`,`OR`, and `NOT` to set very precise access rules. You must follow the bind rules to create valid expressions.

When creating boolean bind rules, always use parentheses to define the order in which rules are to be evaluated. A trailing semicolon is a required delimiter that must appear after the final rule.

For example, to bind with `bindRuleA`, and with either `bindRuleB`, or with either `bindRuleC` and `bindRuleD`, use the following syntax:

```
(bindRuleA and (bindRuleB or (bindRuleC and bindRuleD));)
```

Using another example, the following bind rule is evaluated to be true if the bind DN client is accessed from within the `example.com` domain and is a member of either the administrators group or both the mail administrators and calendar administrators groups.

```
(dns = "*.example.com" and (groupdn = "ldap:///cn=administrators,dc=example,dc=com" or
(groupdn = "ldap:///cn=mail administrators,dc=example,dc=com" and
groupdn = "ldap:///cn=calendar administrators,dc=example,dc=com"));)
```

The || operator is allowed only in the groupdn bind rule keyword expression. For all other bind rule expressions, the or operator must be used.

9.4 Understanding Bind Rule Syntax

Whether access is allowed or denied depends on whether an ACI's bind rule is evaluated to be true.

The following sections describe the bind rule syntax and the various keywords that can be used to allow or deny access.

- Overview of Bind Rule Syntax
- Defining User Access (userdn Keyword)
- Defining Group Access Using groupdn Keyword
- Defining Access Based on Value Matching Using userattr Keyword
- Understanding How to Define Access From a Specific IP Address (ip Keyword)
- Understanding How to Define Access From a Specific Domain Using dns Keyword
- Understanding How to Define Access at a Specific Time of Day or Day of Week Using timeofday and dayofweek Keywords
- Understanding How to Define Access Based on Authentication Method Using authmethod Keyword
- Defining Access Based on a Connection's Security Strength Factor Using ssf Keyword

9.4.1 Overview of Bind Rule Syntax

Bind rules use one of the following patterns:

- *keyword* ="*expression*";
- *keyword*!="*expression*";

where equal (=) indicates that the keyword and expression must match in order for the bind rule to be true, and not equal (!=) indicates that the keyword and expression must not match in order for the bind rule to be true.

The quotation marks ("") around the expression and the delimiting semicolon (;) are required. The expressions you can use depend on the associated keyword.

The timeofday keyword also supports the inequality expressions (<, <=, >, >=). The timeofday keyword is the only keyword that supports these expressions.

The following table lists each keyword and the associated expressions and indicates whether wildcard characters are allowed in the expression.

Keyword	Valid Expressions	Wildcard Allowed?

Defining User Access (`userdn` Keyword)	`ldap:///`*distinguishedName*
	`ldap:///all`
	`ldap:///anyone`
	`ldap:///self`
	`ldap:///parent`
	`ldap:///`*suffix*`??sub?(`*filter*`)`
		Allowed, in DN only

Defining Group Access Using `groupdn` Keyword	`ldap:///`*DN*	Not Allowed
Defining Access Based on Value Matching Using `userattr` Keyword	*attribute*`#`*bindType* or *attribute*`#`*value*	Not Allowed
Understanding How to Define Access From a Specific IP Address (`ip` Keyword)	*IPaddress*	Allowed
Understanding How to Define Access From a Specific Domain Using `dns` Keyword	*DNShostName*	Allowed

Understanding How to Define Access at a Specific Time of Day or Day of Week Using `timeofday` and `dayofweek` Keywords	`sun`
	`mon`
	`tue`
	`wed`
	`thu`
	`fri`
	`sat`
		Not Allowed

Understanding How to Define Access at a Specific Time of Day or Day of Week Using `timeofday` and `dayofweek` Keywords	*hhmm* where *hh* is in the range `00-24` and *mm* is in the range `00-60`	Not Allowed

Understanding How to Define Access Based on Authentication Method Using `authmethod` Keyword	`none`
	`simple`
	`ssl`
	`sasl`
	authenticationMethod
		Not Allowed

Defining Access Based on a Connection's Security Strength Factor Using `ssf` Keyword	0-256	Not Allowed

The following sections provide additional information about the bind rule syntax for each keyword.

9.4.2 Defining User Access (`userdn` Keyword)

In this section you will learn about the procedural information to define user access with the `userdn` keyword.

It contains the following topics:

9.4.2.1 About `userdn` Keyword

User access is defined using the `userdn` keyword. The `userdn` keyword requires one or more valid distinguished names in the following format:

```
userdn = "ldap:///dn [|| ldap:///dn]..."

userdn!= "ldap:///dn [|| ldap:///dn]..."
```

where *dn* can be a DN or one of the expressions `anyone`, `all`, `self`, or `parent`. These expressions refer to the following users:

`userdn = "ldap:///anyone"`
Both anonymous and authenticated users

`userdn = "ldap:///all"`
Only authenticated users

`userdn = "ldap:///self"`
Only the same user as the target entry of the ACI

`userdn = "ldap:///parent"`
Only the parent entry of the ACI target

The `userdn` keyword can also be expressed as an LDAP filter in this form:

```
userdn = ldap:///suffix??sub?(filter)
```

Characters that are syntactically significant for a DN, such as commas, must be escaped with a single backslash (`\`).

9.4.2.2 Defining General Access Using `all` Keyword

You can use bind rules to indicate that a permission applies to anyone who has successfully bound to the directory. The `all` keyword therefore allows access by all authenticated users. This allows general access while preventing anonymous access.

For example, to grant read access to the entire tree to all authenticated users, create the following ACI on the `dc=example,dc=com` node:

```
aci: (version 3.0; acl "all-read"; allow (read)
userdn="ldap:///all";)
```

9.4.2.3 Defining Anonymous Access Using `anyone` Keyword

Granting anonymous access to the directory means that anyone can access it without providing a bind DN or password, regardless of the circumstances of the bind. You can limit anonymous access to specific types of access (for example, access for read or access for search) or to specific subtrees or individual entries within the directory. Anonymous access using the `anyone` keyword also allows access by any authenticated user.

For example, to allow anonymous read and search access to the entire `example.com` tree, create the following ACI on the `dc=example,dc=com` node:

```
aci: (version 3.0; acl "anonymous-read-search";
allow (read, search) userdn = "ldap:///anyone";)
```

9.4.2.4 Defining Self Access Using `self` Keyword

Specifies that users are granted or denied access to their own entries. In this case, access is granted or denied if the bind DN matches the DN of the targeted entry. For example, to grant all users in the `example.com` tree write access to their `userPassword` attribute, create the following ACI on the `dc=example,dc=com` node.

```
aci: (targetattr = "userPassword") (version 3.0; acl
"modify own password"; allow (write) userdn = "ldap:///self";)
```

9.4.2.5 Defining Parent Access Using `parent` Keyword

Specifies that users are granted or denied access to the entry only if their bind DN is the parent of the targeted entry. For example, to allow users to modify any child entries of their bind DN, create the following ACI on the `dc=example,dc=com` node:

```
aci: (version 3.0; acl "parent access";
allow (write) userdn="ldap:///parent";)
```

9.4.2.6 Specifying Users With LDAP URLs

You can dynamically target users in ACIs using a URL with a filter as shown in the following example:

```
userdn = "ldap:///suffix??sub?(filter)"
```

For example, all users in the accounting and engineering branches of the `example.com` tree would be granted or denied access to the targeted resource dynamically based on the following URL:

```
userdn = "ldap:///dc=example,dc=com??sub?(|(ou=eng)(ou=acct))"
```

Do not specify a host name or port number within the LDAP URL. LDAP URLs always apply to the local directory server.

9.4.2.7 Specifying Users With Wildcards

You can also specify a set of users by using the wildcard character (`*`). For example, specifying a user DN of `uid=b*,dc=example,dc=com` indicates that only users with a bind DN beginning with the letter `b` is allowed or denied access based on the permissions you set.

9.4.2.8 Specifying Users With a Logical OR of LDAP URLs

Specify several LDAP URLs or keyword expressions to create complex rules for user access. For example:

```
userdn = "ldap:///uid=b*,c=example.com ||
ldap:///cn=b*,dc=example,dc=com";
```

The bind rule is evaluated to be true for users binding with either of the DN patterns.

9.4.2.9 Excluding Specific LDAP URLs

Use the not-equal (`!=`) operator to define user access that excludes specific URLs or DNs. For example:

```
userdn != "ldap:///uid=*,ou=Accounting,dc=example,dc=com";
```

The bind rule is evaluated to be true if the client is not binding as a UID-based distinguished name in the accounting subtree. This bind rule makes sense only if the targeted entry is not under the accounting branch of the directory tree.

9.4.3 Defining Group Access Using `groupdn` Keyword

Members of a specific group can access a targeted resource. This is known as group access. Group access is defined using the `groupdn` keyword to specify that access to a targeted entry is granted or denied if the user binds using a DN that belongs to a specific group.

The following topics describe how to define group access using `groupdn` keyword:

- About `groupdn` Keyword
- Specifying a Group With a Single LDAP URL
- Specifying a Group With a Logical OR of LDAP URLs

9.4.3.1 About `groupdn` Keyword

The `groupdn` keyword requires the distinguished name of one or more groups in the following format:

```
groupdn="ldap:///groupDN [|| ldap:///groupDN]..."
```

The bind rule is evaluated to be true if the bind DN belongs to a group specified by any of the group DNs. The following section give examples using the `groupdn` keyword.

Characters that are syntactically significant for a DN, such as commas, must be escaped with a single backslash (`\`).

9.4.3.2 Specifying a Group With a Single LDAP URL

To specify a group with a single LDAP URL, use the following format:

```
groupdn = "ldap:///cn=Administrators,dc=example,dc=com";
```

The bind rule is evaluated to be true if the bind DN belongs to the Administrators group. For example, to grant the Administrators group permission to write to the entire directory tree, create the following ACI on the `dc=example,dc=com` node:

```
aci: (version 3.0; acl "Administrators-write"; allow (write)
groupdn="ldap:///cn=Administrators,dc=example,dc=com";)
```

9.4.3.3 Specifying a Group With a Logical OR of LDAP URLs

To specify a group with a logical OR of LDAP URL, use the following format:

```
groupdn = "ldap:///cn=Administrators,dc=example,dc=com ||
ldap:///cn=Mail Administrators,dc=example,dc=com";
```

The bind rule is evaluated to be true if the bind DN belongs to either the Administrators or the Mail Administrators group.

9.4.4 Defining Access Based on Value Matching Using `userattr` Keyword

The `userattr` keyword can be used to specify which attribute values must match between the entry used to bind (bind entry) and the targeted entry.

A `userattr` expression has two formats, a bind-type format and an attribute-value format.

The following sections describe how to define access based on value matching:

- Overview of Bind-Type Format
- Overview of Attribute-Value Format
- Example for `USERDN` Bind Type
- Example for `GROUPDN` Bind Type
- Example for `LDAPURL` Bind Type
- Example for Attribute Value
- About Inheritance Level
- Example for Inheritance
- Adding Permissions to a User

9.4.4.1 Overview of Bind-Type Format

This format is named the bind-type format because it uses the bind DN and possibly the bind entry when evaluating a match. It is the more complicated of the two formats. The bind-type format can be used in the following three ways:

- Treat a target entry attribute value as a DN that must match the bind DN
- Treat a target entry attribute value as a group DN that the bind DN must be a member of
- Require that both the bind DN and the bind entry match an LDAP URL specified in a target entry attribute value

The bind-type `userattr` format uses this syntax:

```
userattr = "attrName#bindType"
```

where:

attrName
Is the name of the attribute in the target entry.

bindType
Must be one of the following:

- `USERDN` — The value of *attrName* must match the bind DN.
- `GROUPDN` — The value of *attrName* is a group that must contain the bind DN.
- `LDAPURL` — The value of *attrName* is a URL that is treated as a search that the bind DN and entry must match. To satisfy the search, the URL's *dn* value is used as a base DN that the bind DN must match or have as a parent DN. The URL's *scope* value restricts how far below the base DN the bind DN can match. Finally, the bind entry must match the URL's *filter* value.

The bind type `userattr` format has a special parent keyword that allows targeting of entries levels below the current target entry. See About Inheritance Level for more information about this keyword.

9.4.4.2 Overview of Attribute-Value Format

The attribute-value format requires the following two conditions to match:

- An attribute specified in the `userattr` expression must exist in both the target and bind entries.
- The values of both of these attributes must match a string value specified in the `userattr` expression. This string value cannot be one of the bind type keywords (`USERDN`, `GROUPDN`, `LDAPURL`).

The attribute value `userattr` format uses this syntax:

```
userattr = "attrName#attrValue"
```

where:

attrName
The name of the attribute in both the target and bind entries.

attrValue
The string representing the attribute value (not `USERDN`, `GROUPDN` or `LDAPURL`).

9.4.4.3 Example for `USERDN` Bind Type

The following example of a bind rule `userattr` keyword expression specifies a match between the bind DN and the value of the target entry attribute `manager`.

```
userattr = "manager#USERDN"
```

This bind rule is evaluated to be true if the bind DN matches the value of the `manager` attribute in the target entry. The `manager` attribute in the target entry must match the bind DN. Wildcards are not allowed.

The following example ACI grants a manager full access to all user attributes of entries located in the subtree under the DN `dc=example, dc=comm`:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr="*")
(version 3.0;acl "manager all access";
allow (all) userattr = "manager#USERDN";)
```

9.4.4.4 Example for GROUPDN Bind Type

This is an example of a bind rule `userattr` keyword expression specifying an attribute that contains a group DN that the bind DN must be a member of.

```
userattr = "owner#GROUPDN"
```

The bind rule is evaluated to be true if the bind DN is a member of the group specified in the `owner` attribute of the target entry.

9.4.4.5 Example for LDAPURL Bind Type

This is an example of a bind rule `userattr` keyword expression specifying an attribute that contains an LDAP URL that is treated as a search that the bind DN and entry must match.

```
userattr = "aciurl#LDAPURL"
```

The attribute `aciurl` is an example only.

The bind rule is evaluated to true if the bind DN and bind entry satisfy all of the search requirements specified in the LDAP URL. For example, if the value of `aciurl` is `ldap:///dc=example,dc=com??one?(cn=joe*)`, then the bind DN must satisfy a one-level search under the base DN of `dc=example,dc=com` and the bind entry must satisfy the filter `(cn=joe*)`.

9.4.4.6 Example for Attribute Value

The following example of the bind rule `userattr` keyword expression specifies an attribute value that both the bind entry and target entry must match.

```
userattr = "favoriteBeverage#Water"
```

The bind rule is evaluated to be true if the bind and target entries include the `favoriteBeverage` attribute with a value of `Water`.

9.4.4.7 About Inheritance Level

When you use the `userattr` keyword to associate the entry used to bind with the target entry, the ACI applies only to the target specified and not to the entries below it. In some circumstances, you might want to extend the application of the ACI several levels below the targeted entry. This is possible by using the parent keyword and specifying the number of levels below the target that should inherit the ACI.

When you use the `userattr` keyword in association with the parent keyword, the syntax is as shown in the following example:

```
userattr = "parent[[inheritanceLevel].attribute#bindType"
```

where:

- *inheritanceLevel* is a comma-separated list that indicates how many levels below the target inherit the ACI. You can include ten levels [0,1,2,3,4,..,9] below the targeted entry. Zero (0) indicates the targeted entry.
- *attribute* is the attribute targeted by the `userattr`.
- *bindType* can be either `USERDN` or `GROUPDN`. The `LDAPURL` bind type is not supported with inheritance.

For example, the `userattr = "parent[[0,1].manager#USERDN"` bind rule is evaluated to be true if the bind DN matches the `manager` attribute of the target entry. Also, the bind rule is evaluated to be true for all entries immediately below the target entry (one level below the target) that have `manager` attributes matching the bind DN.

9.4.4.8 Example for Inheritance

The following example indicates that user `bjensen` is allowed to read and search the `cn=Profiles` entry as well as the first level of child entries, which includes `cn=mail` and `cn=news`.

```
cn=Profiles
aci:(targetattr="*")(version 3.0, acl "profiles access" allow(read, search)
userattr="parent[[0,1].owner#USERDN;)
owner=cn=bjensen, ou=people, dc=example, dc=com
cn=mail, cn=Profiles
mailuser: bjensen
cn=news, cn=Profiles
newuser: bjensen
```

If inheritance were not used in this example, you would need to do one of the following:

- Explicitly set read and search access for user `bjensen` on the `cn=Profiles`, `cn=mail`, and `cn=news` entries in the directory.
- Add the owner attribute and the following ACI to the `cn=mail`, `cn=Profiles` and `cn=news,cn=Profiles` entries:

  ```
  aci: (targetattr="*") (version 3.0; acl "profiles access"; allow
  (read,search) userattr="owner#USERDN";)
  ```

9.4.4.9 Adding Permissions to a User

If you use the `userattr` keyword with all or add permissions, you might find that the behavior of the directory server is not what you expect. Typically, when a new entry is created in the directory, the directory server evaluates access rights on the entry being created, and not on the parent entry. However, for ACIs using the `userattr` keyword, this behavior could create a security hole, so the directory server's normal behavior is modified to avoid it.

Consider the following example ACI:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr="*")
(version 3.0; acl "manager-write"; allow (all)
userattr = "manager#USERDN";)
```

This ACI grants managers all rights on the entries of employees that report to them. However, because access rights are evaluated on the entry being created, this type of ACI would also allow any employee to create an entry in which the manager attribute is set to their own DN. For example, disgruntled employee Joe, `cn=Joe,ou=eng,dc=example,dc=com`, might want to create an entry in the Human Resources branch of the tree to use (or misuse) the privileges granted to Human Resources employees.

He could do this by creating the following entry:

```
dn: cn= Trojan Horse,ou=Human Resources,dc=example,dc=com
objectclass: top
...
cn: Trojan Horse
manager: cn=Joe,ou=eng,dc=example,dc=com
```

To avoid this type of security threat, the ACI evaluation process does not grant add permission at *level 0*, that is, to the entry itself. You can, however, use the `parent` keyword to grant add rights below existing entries. You must specify the number of levels below the parent for add rights. For example, the following ACI allows child entries to be added to any entry in the `dc=example,dc=com` that has a manager attribute that matches the bind DN:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr="*")
(version 3.0; acl "parent-access"; allow (add)
userattr = "parent[1].manager#USERDN";)
```

This ACI ensures that add permission is granted only to users whose bind DN matches the `manager` attribute of the parent entry.

9.4.5 Understanding How to Define Access From a Specific IP Address (`ip` Keyword)

Using bind rules, you can indicate that the bind operation must originate from a specific IP address. This is often used to force all directory updates to occur from a given machine or network domain.

The LDIF syntax for setting a bind rule based on an IP address is shown in the following examples:

```
ip = "IPaddressList"
ip != "IPaddressList"
```

The *IPaddressList* is a list of one or more comma-separated elements from among any of the following:

- A specific IPv4 address, such as `123.45.6.7`
- An IPv4/CIDR-compliant address, such as `192.168.0.0/16`
- An IPv4 address with wildcards to specify a subnetwork, such as `12.3.45.*`
- An IPv4 address or subnetwork with a subnetwork mask, such as `123.45.6.*+255.255.255.192`
- An IPv6 address in any of its legal forms and contained in square brackets `[`and`]`, as defined by RFC 2373 (`http://www.ietf.org/rfc/rfc2373.txt`) and RFC 2732 (`http://www.ietf.org/rfc/rfc2732.txt`). The following addresses are equivalent:
  - `[12AB:0000:0000:CD30:0000:0000:0000:0000]`
  - `[12AB::CD30:0:0:0:0]`
  - `[12AB:0:0:CD30::]`
- An IPv6 address with a subnet prefix length, such as `[12AB::CD30:0:0:0:0]/60`

The bind rule is evaluated to be true if the client accessing the directory is located at the named IP address, which can be useful for allowing certain kinds of directory access only from a specific subnet or machine.

> **Note:**
>
> The IP address from which a user authenticates can be spoofed, and can therefore not be trusted. Do not base ACIs on this information alone.

9.4.6 Understanding How to Define Access From a Specific Domain Using `dns` Keyword

A bind rule can specify that the bind operation must originate from a particular domain or host machine. This is often used to force all directory updates to occur from a given machine or network domain.

The LDIF syntax for setting a bind rule based on the DNS host name is as shown here:

```
dns = "DNShostname"
dns != "DNShostname"
```

> **Caution:**
>
> The `dns` keyword requires that the naming service used on your machine is DNS. If the naming service is not DNS, use the `ip` keyword instead.

The `dns` keyword requires a fully qualified DNS domain name. Granting access to a host without specifying the domain creates a potential security threat. For example, the following expression is allowed but not recommended:

```
dns = "legend.eng";
```

You should use a fully qualified name such as:

```
dns = "legend.eng.example.com";
```

The `dns` keyword allows wildcards. For example:

```
dns = "*.example.com";
```

The bind rule is evaluated to be true if the client accessing the directory is located in the named domain. This can be useful for allowing access only from a specific domain.

> **Note:**
>
> Wildcards do not work if your system uses a naming service other than DNS. In this case, if you want to restrict access to a particular domain, then use the `ip` keyword, as described in Understanding How to Define Access From a Specific IP Address (`ip` Keyword).

9.4.7 Understanding How to Define Access at a Specific Time of Day or Day of Week Using `timeofday` and `dayofweek` Keywords

You can use bind rules to specify that binding can only occur at a certain time of day or on a certain day of the week. For example, you can set a rule that allows access only if the time is between the hours of 8 a.m. and 5 p.m. Monday through Friday. The time used to evaluate access rights is the time on the directory server, not the time on the client.

The LDIF syntax for setting a bind rule based on the time of day is as shown here:

```
timeofday operator "time"
```

where *operator* can be one of the following symbols:

- = (equal to)
- != {not equal to}
- > (greater than)
- >= (greater than or equal to)
- < (less than)
- <= (less than or equal to)

The time is expressed as four digits representing hours and minutes in the 24-hour clock (*hhmm* where *hh* is in the range 00-24 and *mm* is in the range 00-60). For example:

- `timeofday = "1200";` is true if the client is accessing the directory during the minute that the system clock shows noon.
- `timeofday!= "0100";` is true for access at any other time than 1 a.m.
- `timeofday> "0800";` is true for access from 8:01 a.m. through 11:59 p.m.
- `timeofday>= "0800";` is true for access from 8:00 a.m. through 11:59 p.m.
- `timeofday< "1800";` is true for access from 12:00 midnight through 5:59 p.m.

The time and date on the directory server are used for the evaluation of the `timeofday` and `dayofweek` bind rules and not the time on the client.

The LDIF syntax for setting a bind rule based on the day in the week is as shown here:

```
dayofweek = "day1, day2 ..."
```

The possible values for the `dayofweek` keyword are the English three-letter abbreviations for the days of the week: `sun`, `mon`, `tue`, `wed`, `thu`, `fri`, `sat`. Specify all days you want to grant access, for example:

```
dayofweek = "mon, tue, wed, thu, fri";
```

The bind rule is true if the directory is being accessed on one of the days listed.

9.4.8 Understanding How to Define Access Based on Authentication Method Using `authmethod` Keyword

You can set bind rules that state that a client must bind to the directory using a specific authentication method.

The following authentication methods are available:

`None`
Authentication is not required. This is the default. It represents anonymous access.

`Simple`
The client must provide a user name and password to bind to the directory.

SSL
The client must bind to the directory over a Secure Sockets Layer (SSL) or Transport Layer Security (TLS) connection.
For SSL, the connection is established to the LDAPS second port. For TLS, the connection is established through a Start TLS operation. In both cases, you must provide a certificate. For information about setting up SSL, see Using SASL Authentication.

SASL
The client must bind to the directory using a Simple Authentication and Security Layer (SASL) mechanism, such as DIGEST-MD5 or GSSAPI.

The LDIF syntax for setting a bind rule based on an authentication method is as shown here:

```
authmethod = "authentication_method"
```

where `authentication_method` is `none`, `simple`, `ssl`, or `sasl sasl_mechanism`.

The following examples show typical specifications of the `authmethod` keyword:

`authmethod = "none"`
Authentication is not checked during bind rule evaluation.

`authmethod = "simple"`
The bind rule is evaluated to be true if the client is accessing the directory using a user name and password.

`authmethod = "ssl"`
The bind rule is evaluated to be true if the client authenticates to the directory using a certificate over LDAPS. It is not true if the client authenticates using simple authentication (bind DN and password) over LDAPS.

`authmethod = "sasl DIGEST-MD5"`
The bind rule is evaluated to be true if the client is accessing the directory using the SASL DIGEST-MD5 mechanism. Other supported SASL mechanisms are EXTERNAL and GSSAPI.

9.4.9 Defining Access Based on a Connection's Security Strength Factor Using `ssf` Keyword

In this section you will learn how to define access based on a connection's security strength factor keyword.

It contains the following topics:

- Overview of Bind Rule Using Security Strength Factor
- DIGEST-MD5 QOP Key Size Mapping
- TLS Cipher Key Size Mapping
- Example of Using SSF Strength

9.4.9.1 Overview of Bind Rule Using Security Strength Factor

You can use bind rules to specify that binding can only occur based on a specific level of Security Strength Factor (SSF) enforced on the established connection. A connection's SSF is based on the key strength of the cipher enforced on the connection and pertains only to TLS/SSL or DIGEST-MD5/GSSAPI confidentiality or integrity connections.

The LDIF syntax for setting a bind rule based on the Security Strength Factor is shown here:

`ssf` *operator* `"`*strength*`"`

where *operator* can be one of the following symbols:

- `=` (equal to)
- `!=` (not equal to)
- `>` (greater than)
- `>=` (greater than or equal to)
- `<` (less than)
- `<=` less than or equal to

The strength is a value representing the cipher key strength required on the connection and is a value (`0` to `256`). DIGEST-MD5/GSSAPI connections with integrity enforced have an SSF of `1`. TLS/SSL and DIGEST-MD5/GSSAPI confidentiality connections can have variable values of SSF based on the cipher negotiation performed between the directory server and client. The higher a connection's negotiated SSF is, the stronger the encryption is on the connection, as shown in these examples:

- `ssf = "1";` is true for access if integrity `ssf = 1` only is enforced on the connection.
- `ssf!= "40";` is true for access if `ssf not equal 40` is enforced on the connection.
- `ssf> "128";` is true for access if `ssf greater than 128` is enforced on the connection.
- `ssf>= "128";` is true for access if `ssf greater than or equal 128` is enforced on the connection.
- `ssf< "56";` is true for access if `ssf less than 56` is enforced on the connection.

Clear connections have an SSF of `0`.

9.4.9.2 DIGEST-MD5 QOP Key Size Mapping

The following table illustrates the Quality of Protection (QOP) to cipher key size mapping.

Cipher	QOP	Description
RC4 (40)	Low	RC4 cipher with 40-bit key (obsolete)
RC4 (56)	Medium	RC4 cipher with 56-bit key
DES	Medium	Data Encryption Standard (DES) cipher in cipher block chaining (CBC) mode with a 56-bit key
RC4 (128)	High	RC4 cipher with 128-bit key
Triple DES	High	Triple DES cipher in CBC mode with EDE with the same key for each E stage (also called "two keys mode") for a total key length of 112 bits

9.4.9.3 TLS Cipher Key Size Mapping

The following table illustrates the TLS RFC to cipher key size mapping.

Cipher	TLS RFC	Key Size	Description
RC2_CBC_40	4346	40	RC2 cipher in cipher block chaining (CBC) mode (obsolete)
RC4_40	4346	40	RC4 cipher (obsolete)

Cipher	TLS RFC	Key Size	Description
DES40_CBC	4346	40	DES 40-bit cipher in cipher block chaining (CBC) mode (obsolete)
DES_CBC	4346	56	DES 56-bit in cipher block chaining (CBC) mode cipher
3DES_EDE_CBC	4346	112	TDES
RC4_128	4346	128	RC4 cipher
IDEA_CBC	4346	128	International Data Encryption Algorithm (IDEA) cipher in cipher block chaining (CBC) mode
SEED_CBC	4162	128	SEED cipher in cipher block chaining (CBC) mode
CAMELLIA_128_CBC	4132	128	Camellia cipher in cipher block chaining (CBC) mode
AES_128_CBC	3268	128	Advanced Encryption Standard (AES) in cipher block chaining (CBC) mode
AES_256_CBC	3268	256	Advanced Encryption Standard (AES) in cipher block chaining (CBC) mode
CAMELLIA_256_CBC	4132	256	Camellia cipher in cipher block chaining (CBC) mode
AES_256_GCM	5288	256	AES in Galois Counter Mode (GCM)

9.4.9.4 Example of Using SSF Strength

The following ACI allows users to change their own passwords only over a connection with an SSF strength equal to or greater than 128:

```
(targetattr="userPassword||authPassword")(version 3.0; acl "User change pwd";
(allow (write) userdn="ldap:///self" and ssf >= "128");)
```

9.5 Compatibility With the Oracle Directory Server Enterprise Edition Access Control Model

It's time that we delve into the differences between the Oracle Unified Directory access control model with the access control model provided with Oracle Directory Server Enterprise Edition.

This section contains the following topics:

- Global Access Control Instructions
- About the Distinguished Name (DN) Wildcard Matching
- About the Impact of Privilege Subsystem
- About `targetscope` Keyword
- About LDAP Modify Increment Extension
- About Macro Support
- About `roledn` Keyword

9.5.1 Global Access Control Instructions

Global ACI configuration differs from the Oracle Directory Server Enterprise Edition global ACI implementation in some ways. You will learn about those differences in this section.

The Global ACI implementation differs in two ways from Oracle Directory Server Enterprise Edition:

- The `ds-config-global-aci` attribute specifies a global ACI in the `cn=Access Control Handler,cn=config` entry (see Understanding Access Control Principles) rather than placing the ACI in the root DSE entry.
- The scope of the global ACI can be narrowed by specifying a target keyword in the ACI. For example, the following global ACI restricts anonymous read access to entries under the suffix `dc=example,dc=com`:

```
ds-cfg-global-aci: (target="dc=example,dc=com")
(targetattr!="userPassword||authPassword")
(version 3.0; acl "Anonymous read access only under dc=example,dc=com suffix";
allow (read,search,compare) userdn="ldap:///anyone";)
```

Removing the `(target="dc=example,dc=com")` expression would make the ACI global to all entries in Oracle Unified Directory.

9.5.2 About the Distinguished Name (DN) Wildcard Matching

Wild cards appear as asterisk characters and are used in the expression for the target keyword. The asterisk matches an attribute value, a substring of a value, or a DN component.

The ACI DN wildcard matching implementation supports the following usage:

- Any number of wildcards can appear in Relative Distinguished Name (RDN) attribute values, where they match zero or more characters (similar to substring filters). For example, the bind rule matches the following DNs: `uid=bob jensen,dc=example,dc=com` and `uid=bjensen,dc=example,dc=com`:

  ```
  userdn="ldap:///uid=b*jensen*,dc=example,dc=com"
  ```

  It does not match the DN `cn=bill jensen,dc=example,dc=com` because the attribute type of the first RDN does not match.
- A single wildcard can also be used to match any RDN attribute type. (The wildcard in this case can be omitted as a shorthand). For example, these two bind rules behave exactly the same:

  ```
  userdn="ldap:///*=bjensen, dc=example, dc=com"
  userdn="ldap:///bjensen, dc=example, dc=com"
  ```

  They both match the following DNs: `uid=bjensen,dc=example,dc=com` and `cn=bjensen,dc=example,dc=com`.
- A single wildcard can be used to match exactly one RDN component, which can be single or multivalued). For example, the following bind rule matches the DNs `uid=jensen,dc=example,dc=com` and `cn=smith,dc=example,dc=com`:

  ```
  userdn="ldap:///*,dc=example,dc=com"
  ```
- A double wildcard can be used to match one or more RDN components. For example, the following bind rule matches the DNs `uid=jensen,ou=people,dc=example,dc=com` and `uid=jensen,ou=sales,ou=people,dc=example,dc=com`:

```
userdn="ldap:///uid=bjensen,**,dc=example,dc=com"
```

9.5.3 About the Impact of Privilege Subsystem

The Privilege Subsystem allows you to assign refined privileges to users who might require only a specific set of root user access privileges.Oracle Directory Server Enterprise Edition does not support privileges.

The privilege subsystem (discussed in Root Users and the Privilege Subsystem) impacts ACIs in two ways:

- Users with `ds-privilege-name: bypass-acl` privileges can bypass access control evaluation.
- Users needing to modify access control rules need the `ds-privilege-name: modify-acl` privilege.

> **Note:**
>
> Use of the Lightweight Directory Access Protocol (LDAP) Proxied Authorization Control (`http://www.ietf.org/rfc/rfc4370.txt`) requires the bind user to have the `ds-privilege-name: proxied-auth` privilege. When the proxied authorization control is used, evaluation of the `ds-privilege-name: bypass-acl` privilege is performed using the bind user, not the proxied user.
>
> In general, a user should not have both the `ds-privilege-name: proxied-auth` and `ds-privilege-name: bypass-acl` privileges simultaneously since this allows a proxied user to bypass ACI access evaluation.

9.5.4 About `targetscope` Keyword

The `targetscope` keyword differs from Oracle Directory Server Enterprise Edition by including a new scope.

`subordinate`
Restricts the ACI to the subtree below the target resource only.

9.5.5 About LDAP Modify Increment Extension

Oracle Unified Directory supports the LDAP Modify-Increment Extension.

This extension is not supported in Oracle Directory Server Enterprise Edition. Attributes that are to be incremented must have write permissions. For more information, see `https://www.ietf.org/rfc/rfc4525.txt`.

9.5.6 About Macro Support

Oracle Unified Directory supports macros in ACIs.

9.5.7 About `roledn` Keyword

Roles are not supported in Oracle Unified Directory, so the `roledn` keyword should not be used. Equivalent functionality can be achieved by using groups.

9.6 Using Macro ACIs for Advanced Access Control

Organizations that use repeating directory tree structures can enhance the performance and ACI memory usage by using macros to optimize the number of ACIs in the directory tree.

When you reduce the number of ACIs in your directory tree, it is easier to manage your access control policy.

This section describes macro ACIs and its usage, and contains the following topics:

9.6.1 What are Macros?

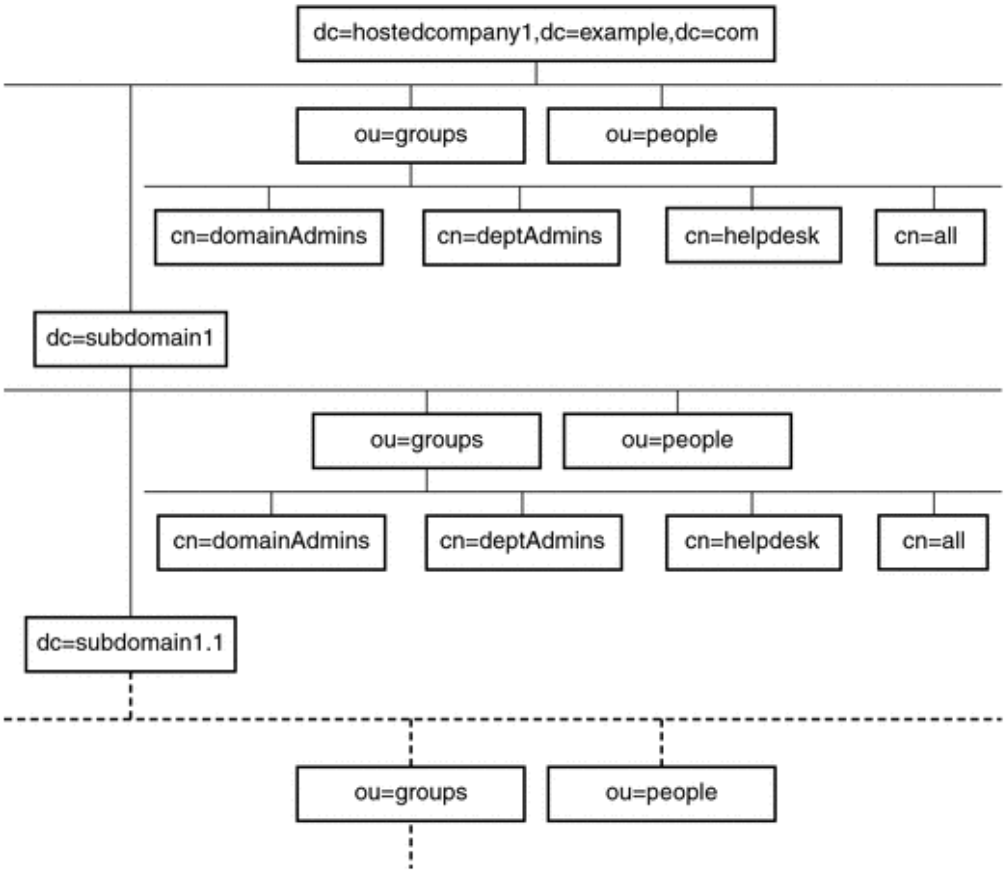
Macros are placeholders used to represent a DN or a part of a DN in an ACI. You can use a macro to represent a DN in the target section of the ACI, in the bind rule section, or in both.

In practice, when Directory Server receives an incoming LDAP operation, the ACI macros are matched against the resource targeted by the LDAP operation. The matching occurs to determine a matching substring, if it exists. If a match exists, the bind rule-side macro is expanded using the matched substring, and access to the resource is determined by evaluating that expanded bind rule.

9.6.2 Example of Macro Access Control Instructions

The advantage of using macro ACIs and how they work are best explained through an example. Figure 9-1 shows a directory tree that uses macro ACIs to effectively reduce the total number of ACIs.

Figure 9-1 Example Directory Tree for Macro ACIs



This illustration uses repeating pattern of subdomains with the same tree structure (`ou=groups, ou=people`). This pattern is also repeated across the tree because the `example.com` directory tree stores the suffixes `dc=hostedCompany2, dc=example,dc=com` and `dc=hostedCompany3,dc=example,dc=com` not shown in the preceding graphic.

The ACIs that apply in the directory tree also have a repeating pattern. For example, the following ACI is located on the `dc=hostedCompany1,dc=example,dc=com node:`

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

This ACI grants read and search rights to the DomainAdmins group to any entry in the `dc=hostedCompany1,dc=example,dc=com` tree.

The following ACI is located on the `dc=hostedCompany1,dc=example,dc=com` node:

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

The following ACI is located on the `dc=subdomain1, dc=hostedCompany1, dc=example, dc=com` node:

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,
 dc=example,dc=com";)
```

The following ACI is located on the `dc=hostedCompany2, dc=example, dc=com` node:

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany2, dc=example,dc=com";)
```

The following ACI is located on the `dc=subdomain1, dc=hostedCompany2, dc=example, dc=com node`:

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany2,
 dc=example,dc=com";)
```

In the preceding four ACIs, the only difference is the DN that is specified in the `groupdn` keyword. By using a macro for the DN, it is possible to replace these ACIs with a single ACI at the root of the tree on the `dc=example,dc=com` node. This macro ACI reads as follows:

```
aci: (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
 (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

The `target` keyword, which was not previously used, is utilized in the new ACI.

In this example, the number of ACIs is reduced from four to one. The real determining factor is the number of repeating patterns you have down and across your directory tree.

9.6.3 Understanding Macro Access Control Instructions

This section provides a description of macro access control instructions and the evaluation mechanism for macro ACIs.

It contains the following topics:

- About Macro Access Control Instructions
- Matching for **`($dn)`** in the Target
- About Macro Matching for ($attr.attrName)

9.6.3.1 About Macro Access Control Instructions

Macro ACIs include the following types of expressions to replace a DN or part of a DN:

- `($dn)`
- `[$dn]`
- (`$attr.`*`attrName`*), where *`attrName`* represents an attribute contained in the target entry

In this section, the ACI keywords used to provide bind credentials, such as userdn, roledn, groupdn, and userattr are collectively called the subject of the ACI. The subject determines to whom the ACI applies.

Table 9-2 lists the macros that can be used to replace specific ACI keywords.

Table 9-2 Macro ACI Keywords

Macro	Description	ACI Keywords
`($dn)`	For matching in the target, and direct substitution in the subject. For example, it will match either target or targetfilter and substitute the matched value into userdn, groupdn, or userattr.	(target, targetfilter) and (userdn, groupdn, userattr)
`[$dn]`	For substituting multiple RDNs that work in subtrees of the subject.	(targetfilter) and (userdn, groupdn, userattr)
`($attr.`*`attrName`*`)`	For substituting the value of the *`attributeName`* attribute from the target entry into the subject.	userdn, groupdn, userattr

The following restrictions apply to macro ACI keywords:

- If you use `($dn)` macro in a subject, then you must define a target that contains `($dn)`.
- If you use `[$dn]` macro in a subject, then you must define a target that contains `($dn)`.
- You can combine both the `($dn)` macro and the `[$dn]` macro with the (`$attr.`*`attrName`*) macro in a subject.

9.6.3.2 Matching for `($dn)` in the Target

The `($dn)` macro in the target of an ACI determines the substitution value by comparing it to the entry targeted by the LDAP request. For example, you have an LDAP request targeted at this entry:

```
cn=all, ou=groups, dc=subdomain1, dc=hostedCompany1, dc=example, dc=com
```

In addition, you have an ACI that defines the target as follows:

```
(target="ldap:///ou=Groups,($dn),dc=example,dc=com")
```

The `($dn)` macro matches with `"dc=subdomain1, dc=hostedCompany1"`. This substring is then used for substitutions in the subject of the ACI.

Substituting ($dn) in the Subject

In the subject of the ACI, the `($dn)` macro is replaced by the entire substring that matches in the target. For example:

```
groupdn="ldap:///cn=DomainAdmins,ou=Groups,($dn),dc=example,dc=com"
```

In this scenario, if the string matching `($dn)` in the target is `dc=subdomain1, dc=hostedCompany1`, then the same string is used in the subject. The subject is then expanded as follows:

```
groupdn="ldap:///
cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"
```

In the targetfilter of the ACI, the `($dn)` macro is replaced with the entire substring that matches in the target. For example:

```
(targetattr="*")(targetfilter=(&(objectClass=nsManagedPerson)
(!(memberOf=cn=ServiceAdministrators,ou=Groups,($dn),o=ace industry,c=us))
```

```
(!(memberOf=cn=Service Help Desk Administrators,ou=Groups,
($dn),o=ace industry,c=us))))
```

The targetfilter becomes:

```
(targetattr="*")(targetfilter=(&(objectClass=nsManagedPerson)
(!(memberOf=cn=ServiceAdministrators,ou=Groups,dc=subdomain1,
dc=hostedCompany1,o=ace industry,c=us))
(!(memberOf=cn=Service Help Desk
Administrators,ou=Groups,dc=subdomain1,dc=hostedCompany1,o=ace industry,c=us))))
```

After the macro has been expanded, Directory Server evaluates the ACI following the normal process to determine whether access is granted.

> **Note:**
>
> Unlike a standard ACI, an ACI that uses macro substitution does not necessarily grant access to the child of the targeted entry. This is because when the child DN is the target, the substitution might not create a valid DN in the subject string.

Substituting [$dn] in the Subject

The substitution mechanism for `[$dn]` is slightly different than for `($dn)`. The DN of the targeted resource is examined several times, each time dropping the left-most RDN component, until a match is found.

Consider a scenario in which you have an LDAP request targeted at the `cn=all,ou=groups, dc=subdomain1,dc=hostedCompany1,dc=example,dc=com` subtree, and the following ACI:

```
aci: (targetattr="*") (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
(version 3.0; acl "Domain access"; allow (read,search)
groupdn="ldap:/cn=DomainAdmins,ou=Groups,[$dn], dc=example,dc=com";)
```

The server proceeds as follows to expand this ACI:

1. The server verifies that the `($dn)` in target matches `dc=subdomain1, dc=hostedCompany1`.
2. The server replaces `[$dn]` in the subject with `dc=subdomain1, dc=hostedCompany1`.

   The resulting subject is `groupdn="ldap:///cn=DomainAdmins,ou=Groups, dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"`. If access is granted, because bind DN is a member of that group, macro expansion stops, and the ACI is evaluated. If bind DN is not a member, the process continues.
3. The server replaces `[$dn]` in the subject with `dc=hostedCompany1`.

   The resulting subject is `groupdn="ldap:///cn=DomainAdmins,ou=Groups, dc=hostedCompany1,dc=example,dc=com"`. Bind DN is again tested for being a member of this group and if it is, the ACI is evaluated fully. However, if bind DN is not a member, macro expansion stops with the last RDN of the matched value, and ACI evaluation is finished for this ACI.

   The advantage of the `[$dn]` macro is that it provides a flexible mechanism to grant domain-level administrators access to all the subdomains in the directory tree. Therefore, the `[$dn]` macro is useful for expressing a hierarchical relationship between domains.

   For example, consider the following ACI:

```
aci: (target="ldap:///ou=*,($dn),dc=example,dc=com") (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain)) (version 3.0;
acl "Domain access"; allow (read,search) groupdn= "ldap:/cn=DomainAdmins,
ou=Groups,[$dn],dc=example,dc=com";)
```

The ACI grants access to the members of `cn=DomainAdmins,ou=Groups, dc=hostedCompany1,dc=example,dc=com` to all of the subdomains under `dc=hostedCompany1`. Thus, an administrator who belongs to that group could access, for example, the subtree `ou=people,dc=subdomain1.1,dc=subdomain1`.

However, at the same time, members of `cn=DomainAdmins,ou=Groups, dc=subdomain1.1` would be denied access to the `ou=people,dc=subdomain1, dc=hostedCompany1` and `ou=people,dc=hostedCompany1` nodes.

9.6.3.3 About Macro Matching for ($attr.attrName)

The `($attr.attrname)` macro is always used in the subject part of an ACI. For example, you could define the following `groupdn`:

```
groupdn = "ldap:/cn=DomainAdmins,ou=($attr.ou),dc=HostedCompany1,dc=example,dc=com"
```

Now, assume that the server receives an LDAP operation that is targeted at the following entry:

```
dn: cn=Babs Jensen,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Babs Jensen
sn: Jensen
ou: Sales ...
```

To evaluate the `groupdn` part of the ACI, the server reads the value of the `ou` attribute stored in the targeted entry. The server then substitutes this value in the subject to expand the macro. In this example, the `groupdn` is expanded as follows:

```
groupdn= "ldap:///cn=DomainAdmins,ou=Sales,dc=HostedCompany1,dc=example,dc=com"
```

Directory Server then evaluates the ACI according to the normal ACI evaluation algorithm.

When the attribute that is named in the macro is multivalued, each value is used in turn to expand the macro. The first value that provides a successful match is used.

9.7 Understanding Virtual Access Control Instructions

In this section you will learn about the principles of the access control mechanism provided with the proxy server.

It contains the following topics:

- About the Virtual Access Control Instructions
- About the Virtual Access Control Instructions Syntax
- About the Virtual Access Control Instructions Configuration Model
- Considerations for Virtual Access Control Instructions Usage

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

9.7.1 About the Virtual Access Control Instructions

Oracle Unified Directory allows virtualization by exposing virtual directory view of data. Therefore, Oracle Unified Directory is responsible for controlling who can access that data, and what parts of the data can be accessed.

To control access to virtual directory view of data, you can define virtual ACIs. When Oracle Unified Directory receives a request on a virtual directory data view, it uses the virtual ACIs, and any authentication information provided by the user, to allow or deny access to the information that is requested.

Virtual ACI allows you to define ACIs that apply at workflow level. This means that you can apply virtual ACIs to workflows containing any kind of workflow elements.

9.7.2 About the Virtual Access Control Instructions Syntax

Virtual ACIs have the same syntax as ACI with some restrictions defined in this section.

For more information about the ACI syntax, see Understanding the Syntax of Access Control Instructions. Only bind rules with following keywords are supported:

- `userdn`
- `ip`
- `dns`
- `timeofday` and `dayofweek`
- `authmethod`
- `ssf`

  This is Security Strength Factor. For more information, see Defining Access Based on a Connection's Security Strength Factor Using `ssf` Keyword.

9.7.3 About the Virtual Access Control Instructions Configuration Model

You can define virtual ACIs for each workflow in the network group. However, each workflow can use or not use virtual ACIs.

The `virtual-aci-mode` property of workflow allows you to specify if virtual ACIs should be used or not. If `virtual-aci-mode` is set to `true`, then all operations handling the ACI attribute manage this attribute as a virtual ACI. Attribute is no longer stored along with user data, but is stored in a specific directory information tree (DIT) known as "`cn=virtual acis`."

For each workflow, you can define the access control group to use using the `access-control-group` property. If the virtual ACI feature is disabled, then the workflow can only use the `Local Backends` access control group. If the virtual ACI feature is enabled, then you can use any access control group.

9.7.4 Considerations for Virtual Access Control Instructions Usage

You must bear in my mind some restrictions while implementing virtual ACIs.

- If you install as directory server, then virtual ACIs are not supported.
- If you install Oracle Unified Directory as proxy server, then you can use virtual ACIs in any supported deployment.
- Virtual ACIs does not support all types of bind rules. For more information about supported bind rules, see About the Virtual Access Control Instructions Syntax.
- Global ACIs apply, if virtual ACIs are enabled.
- You can enable replication of `cn=virtual acis`. To do so, you must ensure that configuration of access control groups are identical on replicated servers.

10

Understanding the Oracle Unified Directory Schema Model

The topics listed below describe schema elements in general and illustrates the ways these schema elements are used in Oracle Unified Directory:

- Overview of Matching Rules
- Overview of Attribute Syntaxes
- Understanding Attribute Types
- Understanding Object Classes
- Understanding Name Forms
- Overview of DIT Content Rules
- Understanding DIT Structure Rules
- Understanding Matching Rule Uses

For instructions on viewing the schema using the `ldapsearch` command, see Managing Attribute Types and Managing Object Classes.

10.1 Overview of Matching Rules

Matching Rules enable you to compare values for the same attribute. From the topics listed below, understand about various Matching Rules available, its description format, and commonly used matching rules.

- Understanding Matching Rules
- Understanding Matching Rule Description Format
- Understanding Commonly Used Matching Rules
- Understanding Relative Time Matching Rules
- Understanding Partial Date Or Time Matching Rules
- Understanding Value Normalization

10.1.1 Understanding Matching Rules

Matching rules are used by Oracle Unified Directory to compare two values for the same attribute and to perform matching operations on them.

There are several different types of matching rules, including:

- Equality Matching Rules

  These matching rules are used to determine whether two values are logically equal to each other. Different implementations of equality matching rules can use different criteria for making this determination (for example, whether to ignore differences in capitalization or deciding which spaces are significant).

- Ordering Matching Rules

  These matching rules are used to determine the relative order for two values, for example, when evaluating greater-or-equal or less-or-equal searches, or when the results need to be sorted.

- Substring Matching Rules

  These matching rules are used to determine whether a given substring assertion matches a particular value. A substring assertion is composed of at least one element from the following sets: at most one `subInitial` element, zero or more `subAny` elements, and at most one `subFinal` element.

- Approximate Matching Rules

  These matching rules are used to determine whether two values are approximately equal to each other. This is frequently based on "sounds like" or some other kind of fuzzy algorithm. Approximate matching rules are not part of the official LDAP specification, but they are included in Oracle Unified Directory for added flexibility.

10.1.2 Understanding Matching Rule Description Format

You can use RFC 4512 format to display matching rules in the `matchingRules` attribute of the schema subentry, and to show the properties that can be associated with a matching rule.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.3 describes the matching rule description format, in Augmented Backus-Naur Form (ABNF). For more information about ABNF, see RFC 4234 (http://www.ietf.org/rfc/rfc4234.txt) and RFC 5234 (http://www.ietf.org/rfc/rfc5234.txt).

The following example shows the definition of the matching rule description format:

```
MatchingRuleDescription = LPAREN WSP
numericoid                 ; object identifier
[ SP "NAME" SP qdescrs ]   ; short names (descriptors)
[ SP "DESC" SP qdstring ]  ; description
[ SP "OBSOLETE" ]          ; not active
SP "SYNTAX" SP numericoid  ; assertion syntax
extensions WSP RPAREN      ; extensions
```

The matching rule description includes these elements:

- numericoid

  The numeric OID is used to uniquely identify the matching rule in Oracle Unified Directory. Every matching rule must have a unique OID.

- NAME

  The name elements are human-readable names assigned to the matching rule that can be used to refer to it in place of the OID. A matching rule is not required to have any human-readable names. If it has only a single name, then it is enclosed in single quotes. If there are multiple names for a matching rule, each is enclosed in single quotes with spaces between the names, and parentheses around the entire set of names.

- DESC

  The description element is a human-readable description for the matching rule. There can be at most one description, and if it is present, it should be enclosed in single quotation marks.

- OBSOLETE

The `OBSOLETE` flag indicates whether this matching rule should be considered available for use. If a matching rule is marked `OBSOLETE`, then it should not be possible to create any new attribute types or matching rule uses that reference this matching rule.

- SYNTAX

  The syntax element identifies the attribute syntax with which the matching rule is associated. This element indicates the acceptable format for values on which the matching rule operates. More information about attribute syntaxes can be found in Overview of Attribute Syntaxes. The syntax OID must be included in all matching rule descriptions.

- extensions

  The extensions for a matching rule can be used to identify other properties for that matching rule that might not be included in the standard definition. Oracle Unified Directory does not currently support any extensions for use in matching rules.

For example, the following is the matching rule description for the standard `caseIgnoreMatch` matching rule:

```
( 2.5.13.2 NAME 'caseIgnoreMatch' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

In this case, the OID is `2.5.13.2`. There is one name, which is `caseIgnoreMatch`. There is no description. The OID of the associated syntax is `1.3.6.1.4.1.1466.115.121.1.15` (which is the Directory String syntax). There are no extensions.

10.1.3 Understanding Commonly Used Matching Rules

There are matching rules defined in LDAP, both in the core protocol specification as well as in other related RFCs and Internet Drafts. Many of these matching rules are defined in RFC 4517.

See (`http://www.ietf.org/rfc/rfc4517.txt`) (LDAP Syntaxes and Matching Rules), in section 4.2, for knowing about the various matching rules defined in RFC 4517. The most commonly used matching rules include:

- caseIgnoreMatch, caseIgnoreOrderingMatch, caseIgnoreSubstringsMatch

  These rules are equality, ordering, and substring matching rules, respectively, that ignore differences in capitalization and also treat multiple consecutive spaces as a single space.

- caseExactMatch, caseExactOrderingMatch, caseExactSubstringsMatch

  These rules are equality, ordering, and substring matching rules, respectively, that treat values in a case-sensitive manner but do treat multiple consecutive spaces as a single space.

- octetStringMatch, octetStringOrderingMatch, octetStringSubstringsMatch

  These rules are equality, ordering, and substring matching rules, respectively, that perform byte-for-byte comparisons of the values, treating them as binary data rather than strings.

- numericStringMatch, numericStringOrderingMatch, numericStringSubstringsMatch

  These rules are equality, ordering, and substring matching rules, respectively, that operate on values that start with a numeric digit, and contain only numeric digits and spaces. Spaces are ignored when performing matching with these matching rules.

- distinguishedNameMatch

  This rule is an equality matching rule that operates on distinguished name (DN) values. It ignores spaces around the commas or semicolons that separate DN components, spaces around plus signs that separate RDN components, and spaces around equal signs that

separate RDN attribute type names from their corresponding values. Differences in capitalization are ignored for attribute type names. Equality matching for attribute values is performed using the equality matching rule for the corresponding attribute type.

- doubleMetaphoneApproximateMatch

  This rule is an approximate matching rule that uses the double metaphone algorithm to perform a "sounds like" comparison.

> **Note:**
>
> This matching rule is not part of any official LDAP specification, but it is included in Oracle Unified Directory for added flexibility

10.1.4 Understanding Relative Time Matching Rules

Oracle Unified Directory provides two matching rules for performing a match on relative dates in generalized time attributes, which are `relativeTimeLTOrderingMatch` and `relativeTimeGTOrderingMatch`.

The two matching rules that perform a match in generalized time attributes are defined below:

```
( 1.3.6.1.4.1.26027.1.4.6
NAME ( 'relativeTimeLTOrderingMatch''relativeTimeOrderingMatch.lt' )
SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )

( 1.3.6.1.4.1.26027.1.4.5
NAME ( 'relativeTimeGTOrderingMatch' 'relativeTimeOrderingMatch.gt' )
SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )
```

The syntax applies to attributes with a `GeneralizedTime` syntax, but it does not take a generalized time string. Instead it takes an offset in the format of [+|-]*number*[*unit*] where:

- +|-

  Specifies a time in the past or future. A positive offset (+) computes a time in the future compared to the current time, and a negative offset (-) computes a time in the past compared to the current time. The default value is positive (+).

- number

  Specifies the number of time units as a positive integer

- unit

  Specifies the time unit as a single letter, `s`, `m`, `h`, `d`, or `w`, for seconds, minutes, hours, days, or weeks

When processing the filter, the server computes the current GMT time, adds the offset and compares the attribute value with the new computed value.

The following example represents `pwdExpirationTime >= (Now + 5 days)`.

```
(pwdExpirationTime:1.3.6.1.4.1.26027.1.4.5:=5d)
```

Similarly, the following example represents `pwdExpirationTime <= (Now + 5 days)`.

```
(pwdExpirationTime:1.3.6.1.4.1.26027.1.4.6:=5d)
```

10.1.5 Understanding Partial Date Or Time Matching Rules

Oracle Unified Directory provides the `partialDateAndTimeMatchingRule` matching rule for performing a substring match on dates in generalized time attributes.

The `partialDateAndTimeMatchingRule` matching rule is defined below:

```
( 1.3.6.1.4.1.26027.1.4.7
NAME 'partialDateAndTimeMatchingRule'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )
```

This matching rule applies to attributes with a `GeneralizedTime` syntax, but the value is not a generalized time. Instead, it specifies a pattern for the date, composed of one or more sequences of an integer followed by a tag. The currently supported tags are `Y`, `M`, `D`, `h`, `m`, and `s`.

The following examples use the attribute `birthDate` (described in http://tools.ietf.org/html/draft-gryphon-ldap-schema-vcard4-00) with the following definition:

```
attributeTypes: ( 1.3.6.1.4.1.33592.1.3.2 NAME 'birthDate'
DESC 'birthday'
EQUALITY generalizedTimeMatch
ORDERING generalizedTimeOrderingMatch
SYNTAX .3.6.1.4.1.1466.115.121.1.24
USAGE userApplications SINGLE-VALUE )
```

For example, the following filter matches all users born on September 21st.

```
(birthDate:1.3.6.1.4.1.26027.1.4.7:=09M21D)
```

As another example, the following filter matches all users born in 1965:

```
(birthDate:1.3.6.1.4.1.26027.1.4.7:=1965Y)
```

The following search operation returns all entries with a birthday the fourteenth day of any month:

```
$ ./ldapsearch -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file \
  -b "dc=example,dc=com" \
  "(birthDate:1.3.6.1.4.1.26027.1.4.7:=14D)" birthDate
```

10.1.6 Understanding Value Normalization

One of the tasks that most matching rules need to perform is value normalization. This is the process of transforming a given value to a form that can be used to compare values efficiently.

In most cases, the normalization process should reduce all logically equivalent values to the same string so that a very simple string comparison can be performed to determine whether the strings are equal. For example, the `caseIgnoreMatch` matching rule typically normalizes values by converting all characters to lowercase and replacing occurrences of multiple consecutive spaces with a single space. A more complicated example is the `distinguishedNameMatch` matching rule, which removes all unnecessary spaces (for example, around commas, equal signs, and plus signs), converts all attribute types to lowercase, and then uses the appropriate matching rules to normalize the attribute values for each RDN component.

In some cases, normalization alone is not sufficient for determining whether two values are logically equivalent — particularly for cases in which the value is transformed, and there can be multiple different transformations for the same value.

10.2 Overview of Attribute Syntaxes

Attribute syntaxes are essentially data type definitions. The syntax for an attribute type indicates the type of data meant to be held by the corresponding values. This can be used to determine whether a particular value is acceptable for a given attribute, as well as to provide information about how Oracle Unified Directory should interact with existing values.

Oracle Unified Directory supports the ability to reject values that violate the associated attribute syntax, and this is the default behavior for the purposes of standards compliance. It is possible to disable this attribute syntax checking completely if necessary, but it is also possible to accept values that violate the associated syntax but log a warning message to Oracle Unified Directory's error log every time this occurs. However, if attributes are allowed to have values that violate their associated syntax, matching operations might not behave as expected with such values. For information about disabling schema checking, see Configuring Schema Checking.

The following sections discuss attribute syntax:

- Understanding the Attribute Syntax Description Format
- About the Commonly Used Attribute Syntaxes
- About the Pattern-Matching Syntax Extension
- About the Enumeration Syntax Extension
- About Substitution Syntax Extension

10.2.1 Understanding the Attribute Syntax Description Format

RFC 4512 describes the attribute syntax description format which includes elements, such as, numericoid, DESC, and extensions.

RFC 4512 (`http://www.ietf.org/rfc/rfc4512.txt`), section 4.1.5 describes the attribute syntax description format, as shown in this example:

```
SyntaxDescription = LPAREN WSP
numericoid                ; object identifier
[ SP "DESC" SP qdstring ] ; description
extensions WSP RPAREN     ; extensions
```

The attribute syntax description includes these elements:

- numericoid

  The numeric OID used to uniquely identify the attribute syntax in Oracle Unified Directory.

- DESC

  An optional description for the syntax. If it is provided, then it must be enclosed in single quotation marks.

- extensions

  An optional set of extensions for the attribute syntax. supports the following extensions:

  – `X_PATTERN`: Specifies that the attribute uses the regular expression syntax. See About the Pattern-Matching Syntax Extension for more information.

  – `X-ENUM`: Specifies that the attribute uses the enumerated syntax. See About the Enumeration Syntax Extension for more information.

  - `X-SUBST`: Specifies that the attribute uses the substitution syntax. See About Substitution Syntax Extension for more information.

The following example shows the attribute syntax description for the standard directory string syntax:

```
( 1.3.6.1.4.1.1466.115.121.1.15 DESC 'Directory String' )
```

In this case, the OID is `1.3.6.1.4.1.1466.115.121.1.15`, and the description is `Directory String`. This example specifies no extensions.

10.2.2 About the Commonly Used Attribute Syntaxes

There are numerous attribute syntaxes defined in LDAP, both in the core protocol specification and in other related RFCs and Internet Drafts. The various attribute syntaxes are defined in RFC 4517 and are explained in this topic.

Many of these attribute syntaxes are defined in RFC 4517 (`http://www.ietf.org/rfc/rfc4517.txt`) (LDAP Syntaxes and Matching Rules) in section 3.3. The most commonly used attribute syntaxes include:

- Directory String

  The Directory String syntax is used to hold general-purpose string values containing one or more UTF-8 characters. Technically, empty values (that is, those with zero characters) are not allowed. Because Oracle Directory Server Enterprise Editionhas historically allowed empty values, Oracle Unified Directoryoffers a configuration option that can be used to allow it as well although it is disabled by default for standards compliance.

- IA5 String

  The IA5 String syntax is used to hold string values based on the IA5 character set, which is also known as the ASCII character set.

- Printable String

  The Printable String syntax is used to hold string values that contain one or more characters from the set of uppercase and lowercase letters, numeric digits, single quotes, left and right parentheses, plus sign, comma, hyphen, period, and equal sign.

- Boolean

  The Boolean syntax is used to hold values of either `TRUE` or `FALSE`. No other values are allowed for attributes with this syntax.

- Integer

  The Integer syntax is used to hold integer values, which must contain at least one digit. It can start with a hyphen to indicate a negative value. Zero can be used as the first digit only when the value is zero.

- Octet String

  The Octet String syntax is used to hold a set of zero or more bytes. It has been used to replace the former Binary syntax.

- DN

  The DN syntax is used to hold distinguished name values, comprised of zero or more RDN components. Values should be in the format specified in RFC 4514 (`http://www.ietf.org/rfc/rfc4514.txt`) (LDAP String Representation of Distinguished Names).

10.2.3 About the Pattern-Matching Syntax Extension

The `X-PATTERN` attribute syntax extension can be used to define new string syntaxes with values restricted by one or more regular expressions.

The following example adds an `X-PATTERN` attribute syntax to the schema.

```
$ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file
dn: cn=schema
changetype: modify
add: ldapsyntaxes
ldapSyntaxes: ( 1.3.6.1.4.1.32473.1 DESC 'Host and Port in the format of HOST:PORT'
  X-PATTERN '^[a-zA-Z][a-zA-Z0-9-]+:[0-9]+$' )
```

This new syntax can be used to define attributes and object classes, as shown in the following example.

```
$ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( 1.3.6.1.4.1.32473.2 NAME 'example-attr-regex' SYNTAX
1.3.6.1.4.1.32473.1 )
-
add: objectclasses
objectclasses: ( 1.3.6.1.4.1.32473.3 NAME 'exampleOCregex' SUP top AUXILIARY MUST
example-attr-regex)
-
```

Values for `example-attr-regex` attributes must match the defined pattern, or the server rejects them. The following attribute fits pattern defined in the example syntax, so the server accepts it:

```
example-attr-regex: localhost:389
```

The following attribute is rejected because it does not include the required colon and numeric string:

```
localhost
```

The following attribute is rejected because it contains periods (.), which are not specified as part of the `HOST` component:

```
host.example.com:389
```

10.2.4 About the Enumeration Syntax Extension

The `X-ENUM` attribute syntax extension can be used to define new string syntaxes with values restricted to a set of defined, ordered values.

The following example defines an `X-ENUM` attribute to the schema.

```
$ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file
dn: cn=schema
changetype: modify
add: ldapsyntaxes
ldapSyntaxes: ( 1.3.6.1.4.1.32473.4 DESC 'Day Of The Week'
  X-ENUM ( 'monday' 'tuesday' 'wednesday' 'thursday'
  'friday' 'saturday' 'sunday' ) )
```

This new syntax can be used to define attributes and object classes, as shown in the following example.

```
$ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( 1.3.6.1.4.1.32473.5 NAME 'example-attr-enum' SYNTAX
1.3.6.1.4.1.32473.4 )
-
add: objectclasses
objectclasses: ( 1.3.6.1.4.1.32473.6 NAME 'exampleOCenum' SUP top AUXILIARY
MUST example-attr-enum)
```

Values for `example-attr-enum` attributes must match the defined pattern, or the server rejects them.

Enumerated values are not case-sensitive, so both of the following examples are accepted:

```
example-attr-enum: thursday
example-attr-enum: Thursday
```

Enumerated attribute values are literal (and not internationalized), so the following example does not match the pattern and is rejected, regardless of any semantic equivalence:

```
example-attr-enum: jeudi
```

The defined values specify an order, so enumerated attributes can be used in relative comparison filters, as shown in the following example:

```
(example-attr-enum>=wednesday)
```

The preceding comparison filter matches, for example, a value of `thursday`. The comparison is based on the order of the enumerated values, and ASCII values are not applicable in this case.

10.2.5 About Substitution Syntax Extension

The `X-SUBST` attribute syntax extension can be used to define new string syntaxes with values in terms of existing syntaxes. It is provided for use when extending the native directory server schema with a nonstandard schema (or an external schema) that uses syntaxes not supported by Oracle Unified Directory. Instead of altering the imported schema, extend it with the `X-SUBST` extension to instruct Oracle Unified Directory to treat values in terms of a supported syntax.

The following example defines a new syntax, `AttCertPath`, in terms of an existing syntax, `1.3.6.1.4.1.1466.115.121.1.15`, directory string. This change must be made under `cn=schema`.

```
$ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file
dn: cn=schema
objectClass: top
objectClass: ldapSubentry
objectClass: subschema
ldapSyntaxes: ( 1.3.6.1.4.1.4203.666.11.10.2.4
  DESC 'AttCertPath'
  X-SUBST '1.3.6.1.4.1.1466.115.121.1.15' )
```

This feature can be useful during migration and can lessen the impact on the schema. For example, during migration to Oracle Unified Directory, an incoming schema can contain attribute definitions that use an undefined syntax. The `X-SUBST` attribute syntax extension provides a means to define those missing syntaxes in terms of other, more general syntaxes.

With this capability, the schema and data can be migrated without the need to modify the schema or data or to implement new syntaxes.

10.3 Understanding Attribute Types

Attribute types define the set of attributes that can be used in Oracle Unified Directory and how operations involving those attributes should be conducted. Among other things, it combines an attribute syntax and set of matching rules with a unique OID and human-readable names.

The following sections describe attribute types:

- Understanding Attribute Type Description Format
- Understanding Attribute Type Inheritance
- About Attribute Type Implementation

10.3.1 Understanding Attribute Type Description Format

Learn about the Attribute Type Descriptions and the various elements included in it.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.2 describes the attribute type description format, as shown here:

```
AttributeTypeDescription = LPAREN WSP
    numericoid                    ; object identifier
    [ SP "NAME" SP qdescrs ]      ; short names (descriptors)
    [ SP "DESC" SP qdstring ]     ; description
    [ SP "OBSOLETE" ]             ; not active
    [ SP "SUP" SP oid ]           ; supertype
    [ SP "EQUALITY" SP oid ]      ; equality matching rule
    [ SP "ORDERING" SP oid ]      ; ordering matching rule
    [ SP "SUBSTR" SP oid ]        ; substrings matching rule
    [ SP "SYNTAX" SP noidlen ]    ; value syntax
    [ SP "SINGLE-VALUE" ]         ; single-value
    [ SP "COLLECTIVE" ]           ; collective
    [ SP "NO-USER-MODIFICATION" ] ; not user modifiable
    [ SP "USAGE" SP usage ]       ; usage
    extensions WSP RPAREN         ; extensions

usage = "userApplications"     /  ; user
        "directoryOperation"   /  ; directory operational
        "distributedOperation" /  ; DSA-shared operational
        "dSAOperation"            ; DSA-specific operational
```

The attribute type description includes these elements:

- numericoid

  The numeric OID used to uniquely identify the attribute type in Oracle Unified Directory. Although the specification requires a numeric OID,Oracle Unified Directory also allows a non-numeric OID for the purpose of convenience and better compatibility with Oracle Unified Directory. In this case, the non-numeric OID should be the same as the name of the attribute type followed by the string `-oid`.

- NAME

  An optional set of human-readable names that can also be used to refer to the attribute type. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If there is a description, then it should be enclosed in single quotation marks.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the attribute type is active. If an attribute type is marked as `OBSOLETE`, then it means that it should not be referenced by any new elements created in Oracle Unified Directory.

- SUP

  An optional reference to the superior attribute type. If there is a superior type, then it may be referenced by either its OID or any of its human-readable names.

- EQUALITY

  An optional equality matching rule definition. If a specific equality matching rule is provided, then it can be referenced by either its OID or any of its human-readable names. If no equality matching rule is given, then the attribute type uses the default equality matching rule for the associated attribute syntax. If the attribute syntax does not have a default equality matching rule, then equality matching operations are not allowed for attributes of that type.

- ORDERING

  An optional ordering matching rule definition. If a specific ordering matching rule is provided, then it can be referenced by either its OID or any of its human-readable names. If no ordering matching rule is given, then the attribute type uses the default ordering matching rule for the associated attribute syntax. If the attribute syntax does not have a default ordering matching rule, then ordering matching operations are not allowed for attributes of that type.

- SUBSTR

  An optional substring matching rule definition. If a specific substring matching rule is provided, then it can be referenced by either its OID or any of its human-readable names. If no substring matching rule is given, then the attribute type uses the default substring matching rule for the associated attribute syntax. If the attribute syntax does not have a default substring matching rule, then substring matching operations are not allowed for attributes of that type.

- SYNTAX

  An optional attribute syntax for use with the attribute type. If it is provided, then it should be given as a numeric OID. The syntax identifier can also optionally contain an integer value enclosed in curly braces directly following the OID (without any spaces between the last digit of the OID and the opening curly brace), which may be used to suggest a minimum upper bound on the length of values for attributes of that type. Oracle Unified Directory does not enforce any maximum length restrictions for attribute values, so if a length is given, then it is ignored.

- SINGLE-VALUE

  An optional `SINGLE-VALUE` flag that indicates that attributes of that type are allowed to have only a single value in any entry in which they appear. If this flag is not present in the attribute type description, then attributes of that type are allowed to have multiple distinct values in the same entry.

- COLLECTIVE

  An optional `COLLECTIVE` flag that indicates that the attributes of that type are assigned their values by virtue in their membership in some collection. Collective attributes are described

in RFC 3671 (http://www.ietf.org/rfc/rfc3671.txt) (Collective Attributes in LDAP) and are one of the types of virtual attributes that are supported in Oracle Unified Directory.

- NO-USER-MODIFICATION

  An optional NO-USER-MODIFICATION flag that indicates that values of attributes of that type cannot be modified by external clients (that is, the values can be modified only by internal processing within Oracle Unified Directory).

- USAGE

  An optional usage specification that indicates how the attribute type is to be used. The following attribute usages are allowed:

  - userApplications — Used to store user data.
  - directoryOperation — Used to store data required for internal processing within Oracle Unified Directory.
  - distributedOperation — Used to store operational data that must be synchronized across servers in the topology.
  - dSAOperation — Used to store operational data that is specific to a particular directory server and should not be synchronized across the topology.

- extensions

  An optional set of extensions for the attribute type.Oracle Unified Directory currently uses the following extensions for attribute types:

  - X-ORIGIN — Provides information about where the attribute type is defined (for example, whether it is defined by a particular RFC or Internet Draft or whether it is defined within the project).
  - X-SCHEMA-FILE — Indicates which schema file contains the attribute type definition.
  - X-APPROX — Indicates which approximate matching rule should be used for the attribute type. If this is specified, then its value should be the name or OID of a registered approximate matching rule.

For example, the following is the attribute type description for the standard uid attribute type:

```
( 0.9.2342.19200300.100.1.1 NAME 'uid' EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256}
X-ORIGIN 'RFC 4519' )
```

In this case, the OID is 0.9.2342.19200300.100.1.1. There is a single human-readable name of uid. The caseIgnoreMatch rule should be used for equality matching, and the caseIgnoreSubstringsMatch rule should be used for substring matching. The attribute type uses the directory string syntax with a suggested minimum upper bound of 256 characters, and the attribute type definition was taken from RFC 4519 (http://www.ietf.org/rfc/rfc4519.txt). There is no description or superior type specified. The attribute type is not marked OBSOLETE, SINGLE-VALUE, COLLECTIVE, or NO-USER-MODIFICATION. There is no ordering matching rule specified, which means that Oracle Unified Directoryfalls back on the default ordering rule used by the directory string syntax. There is no X-APPROX extension to specify the approximate matching rule so the default approximate rule for the directory string syntax is used there as well.

10.3.2 Understanding Attribute Type Inheritance

One attribute type can reference another as its superior type. By doing that it can inherit the matching rule and the attribute syntax from its superior type

This has two primary effects:

- The matching rule and attribute syntax specifications from the superior attribute type can be inherited by the subordinate type if the subordinate does not override the superior definition. For example, if the superior attribute type uses the `IA5 String` syntax, then the subordinate attribute type also uses the `IA5 String` syntax unless its definition overrides that by specifying an alternate syntax. According to the specification in RFC 4512 (`http://www.ietf.org/rfc/rfc4512.txt`), section 2.5.1, an attribute type can have a different syntax than its superior type only if the syntax for the subordinate type is a refinement of (that is, allows a subset of the values of) the syntax for the superior attribute type.
- The OID, any of the human-readable names associated with the superior attribute type, or both can be used to collectively reference all of the subordinate types. For example, the name attribute type is referenced as the superior type for the `cn`, `sn`, `c`, `l`, `st`, `o`, `ou`, `title`, `givenName`, `initials`, `generationQualifier`, and `dmdName` attribute types. Therefore, a filter of `(name=test)` should match an entry if any attribute with one of those types has a value of `test`.

A subordinate attribute type cannot have a different usage than its superior type. That is, if the superior type is `userApplications`, then the subordinate type must also be `userApplications`. Similarly, if a superior type is declared `COLLECTIVE`, then the subtype must also be `COLLECTIVE`, but if the superior type is not `COLLECTIVE`, then the subordinate type must also not be `COLLECTIVE`.

10.3.3 About Attribute Type Implementation

The mechanism that is used to handle attribute types varies from the LDAPv3 specification.

The mechanism varies in the following ways:

- The LDAPv3 specification states that a subordinate attribute type must have the same syntax as the superior type, or a refinement of that syntax. Oracle Unified Directory does not enforce this constraint because it does not have any way to determine whether one attribute syntax is a refinement of the syntax of the supertype.
- The synchronization subsystem does not take attribute usage into account (for example, so that attribute types with a usage of `dSAOperation` are not synchronized).

10.4 Understanding Object Classes

Object classes are essentially named sets of attribute types that can be used to control the type of data that can be stored in entries.

> **Note:**
>
> The terms "object class" and "objectclass" (that is, with and without a space between the words) are generally used interchangeably.

The following sections describe object classes:

- Understanding Object Class Description Format
- About Object Class Kinds
- About Object Class Inheritance
- About Directory Server Object Class Implementation

10.4.1 Understanding Object Class Description Format

Learn about the object class description format and the various elements included in it.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.1 describes the object class description format, as shown here:

```
ObjectClassDescription = LPAREN WSP
numericoid                 ; object identifier
[ SP "NAME" SP qdescrs ]   ; short names (descriptors)
[ SP "DESC" SP qdstring ]  ; description
[ SP "OBSOLETE" ]          ; not active
[ SP "SUP" SP oids ]       ; superior object classes
[ SP kind ]                ; kind of class
[ SP "MUST" SP oids ]      ; attribute types
[ SP "MAY" SP oids ]       ; attribute types
extensions WSP RPAREN
kind = "ABSTRACT" / "STRUCTURAL" / "AUXILIARY"
```

The object class description includes these elements:

- numericoid

  The numeric OID used to uniquely identify the object class in Oracle Unified Directory. Although the specification requires a numeric OID, Oracle Unified Directory also allows a non-numeric OID for the purpose of convenience and better compatibility with the Oracle Unified Directory. In this case, the non-numeric OID should be the same as the name of the object class followed by the string `-oid`.

- NAME

  An optional set of human-readable names that can be used to refer to the object class. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If there is a description, then it should be enclosed in single quotation marks.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the object class is active. If an object class is marked as `OBSOLETE`, then it should not be referenced by any new elements created in Oracle Unified Directory.

- SUP

  An optional set of one or more superior classes for the object class.

> **Note:**
>
> Although, technically, the specification allows an object class to have multiple superior classes, Oracle Unified Directory currently only supports a single superior class.
>
> In this case, the `SUP` keyword should be followed by a space and the name or OID of the superior class. If there are multiple superior classes, then they should be separated by dollar signs and the entire set of superior classes should be enclosed in parentheses.

- kind

  An optional keyword that specifies the kind of object class that is being defined. If this is specified, then it must be one of `ABSTRACT`, `STRUCTURAL`, or `AUXILIARY`. If no value is specified, then the object class is considered `STRUCTURAL`.

- MUST

  An optional set of attribute types for attributes that are required to be present (that is, have at least one value) in entries with that object class. If there is only a single required attribute, then the `MUST` keyword should be followed by the name or OID of that attribute type. If there are multiple required attribute types, then they should be separated by dollar signs and the entire set of required attribute types should be enclosed in parentheses.

- MAY

  An optional set of optional attribute types for attributes that are allowed (but not required) to be present in entries with that object class. If there is only a single optional attribute, then the `MAY` keyword should be followed by the name or OID of that attribute type. If there are multiple optional attribute types, then they should be separated by dollar signs and the entire set of optional attribute types should be enclosed in parentheses.

- extensions

  An optional set of extensions for the object class.Oracle Unified Directory currently uses the following extensions for object classes:

  - `X-ORIGIN` — Provides information about where the object class is defined (for example, whether it came from a particular RFC or Internet Draft or if it is defined within the project).
  - `X-SCHEMA-FILE` — Indicates which schema file contains the object class definition (This extension is generally used for internal purposes only and is exposed to clients.)

For example, the following is the object class description for the standard `person` object class:

```
( 2.5.6.6 NAME 'person' SUP top STRUCTURAL MUST ( sn $ cn )
MAY ( userPassword $ telephoneNumber $ seeAlso $ description )
X-ORIGIN 'RFC 4519' )
```

In this case, the OID is `2.5.6.6`. There is a single human-readable name of `person`. The superior class is `top`. The kind is `STRUCTURAL`. Any entry containing the `person` object class is required to include the `sn` and `cn` attributes and is allowed to include the `userPassword`, `telephoneNumber`, `seeAlso`, and `description` attributes. The object class definition is taken from RFC 4519 (`http://www.ietf.org/rfc/rfc4519.txt`). There is no description, and the object class is not considered `OBSOLETE`.

10.4.2 About Object Class Kinds

Learn about the different kinds of object classes and their similarities with the model used by the Java programming language.

As described in Understanding Object Class Description Format, all object classes must have a kind of either `ABSTRACT`, `STRUCTURAL`, or `AUXILIARY`:

- `ABSTRACT` object classes are intended only to be extended by other object classes. An entry must not contain any abstract class unless it also contains a structural or auxiliary class that derives from that abstract class (that is, includes a non-abstract object class which has the abstract class in its inheritance chain). All entries must contain at least the `top` abstract object class in the inheritance chain for their structural class. They may or may not contain other abstract classes in the inheritance chains for their structural class or any of their auxiliary classes.
- `STRUCTURAL` object classes are intended to define the crux of what an entry represents. Every entry must include exactly one structural object class chain, and the root of that chain must ultimately be the `top` abstract object class. The structural object class for an entry cannot be changed.
- `AUXILIARY` object classes are intended to define additional qualities of entries. An entry can contain zero or more auxiliary classes, and the set of auxiliary classes associated with an entry can change over time.

The model represented by object class kinds translates very neatly to the model used by the Java programming language. Abstract LDAP object classes map directly to Java abstract classes, auxiliary LDAP object classes map directly to Java interfaces, and structural LDAP object classes map directly to Java concrete (non-abstract) classes. Just as Java classes must extend exactly one superclass but can implement any number of interfaces, so must LDAP entries contain exactly one structural class chain but can include any number of auxiliary class chains. Similarly, just as it is not possible to directly instantiate an abstract Java class, it is also not possible to create an LDAP entry containing only abstract object classes.

Oracle Directory Server Enterprise Edition has never enforced many of the restrictions noted here around object class kinds. In particular, it would allow the creation of entries that did not contain any structural object class chain and would also allow the creation of entries containing multiple structural object class chains. This means that deployments using Oracle Directory Server Enterprise Edition can contain entries that violate this constraint.Oracle Unified Directory does not allow this behavior by default, but for the sake of compatibility with existing Oracle Unified Directory deployments, it is possible to configure Oracle Unified Directory to allow entries to violate this constraint, optionally writing a message to Oracle Unified Directory's error log each time this condition is detected. However, if there are entries that do not contain exactly one structural object class, then some schema elements like DIT content rules that depend on this constraint might not work as expected in all cases. To configure Oracle Unified Directory to accept these kinds of schema violations, set the `single-structural-objectclass-behavior` property of the global configuration. For more information, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

10.4.3 About Object Class Inheritance

Learn about inheritance in object classes and the restrictions that exists for different kinds of object class inheritance.

As specified in Understanding Object Class Description Format, object classes can have zero or more superior classes (although currently, Oracle Unified Directory supports at most one

superior class). If an object class references a superior class, then all of the required and optional attributes associated with that superior class are also associated with the subordinate class.

The following restrictions exist for object class inheritance:

- `ABSTRACT` object classes can inherit only from other abstract classes. They cannot be subordinate to structural or auxiliary classes.
- `STRUCTURAL` object classes can inherit only from abstract classes or other structural classes. They cannot be subordinate to auxiliary object classes.
- `AUXILIARY` object classes can inherit only from abstract classes or other auxiliary classes. They cannot be subordinate to structural object classes.
- All `STRUCTURAL` object classes must ultimately inherit from the top abstract object class. The net effect of this is that every entry in Oracle Unified Directory must include the `top` object class and so must also include the `objectClass` attribute type, which is required by `the` top object class).

10.4.4 About Directory Server Object Class Implementation

The mechanism that is used to handle object classes varies from the LDAPv3 specification.

In such mechanisms object classes are allowed to have at most one superior class, whereas the specification allows multiple superior classes in some cases.

10.5 Understanding Name Forms

Name forms can be used to define a mechanism for naming entries in Oracle Unified Directory. In particular, a name form specifies one or more attribute types that must be present in the RDN of an entry with a given structural object class. A name form can also specify zero or more attribute types, which can optionally be present in the RDN.

Each structural object class defined in Oracle Unified Directory schema can be associated with one or more name forms. If a name form is defined for a given structural object class, then the associated name form is enforced for any add or modify DN operations for entries containing that object class. If a structural object class is not associated with a name form, then any attribute type that is allowed to exist in the target entry can be used as a naming attribute type.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.7.2 describes the name form description format, as shown here:

```
NameFormDescription = LPAREN WSP
numericoid                 ; object identifier
[ SP "NAME" SP qdescrs ]   ; short names (descriptors)
[ SP "DESC" SP qdstring ]  ; description
[ SP "OBSOLETE" ]          ; not active
SP "OC" SP oids             ; structural object classes
SP "MUST" SP oids          ; attribute types
[ SP "MAY" SP oids ]       ; attribute types
extensions WSP RPAREN      ; extensions
```

The name form description includes these elements:

- numericoid

  The numeric OID used to uniquely identify the name form in Oracle Unified Directory. Although the specification requires a numeric OID,Oracle Unified Directory also allows a

non-numeric OID for the purpose of convenience. In this case, the non-numeric OID should be the same as the name of the name form followed by the string `-oid`.

- NAME

  An optional set of human-readable names that can be used to refer to the name form. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If a description is present, then it should be enclosed in single quotation marks.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the name form is active. If a name form is marked as `OBSOLETE`, then it should not be in effect within Oracle Unified Directory, nor should it be possible to create any other elements that depend on it.

- OC

  The names or OIDs of the structural object classes with which the name form is associated.

- MUST

  The names or OIDs of one or more attribute types that must be present in the RDN for any entry with the specified structural class. If there is only a single required attribute type, then only its name or OID must be given. If there are multiple required attribute types, then they should be separated by spaces and dollar signs, and the entire set of required attribute types should be enclosed in parentheses.

- MAY

  The names or OIDs of zero or more attribute types that can optionally be present in the RDN for any entry with the specified structural class. If there is only a single optional attribute type, then only its name or OID must be given. If there are multiple optional attribute types, then they should be separated by spaces and dollar signs, and the entire set of optional attribute types should be enclosed in parentheses.

- extensions

  An optional set of extensions for the name form.Oracle Unified Directory currently uses the following extensions for name forms:

  - `X-ORIGIN` — Provides information about where the name form is defined (for example, whether it came from a particular RFC or Internet Draft or whether it is defined within the project.).
  - `X-SCHEMA-FILE` — Indicates which schema file contains the name form definition (This extension is generally used for internal purposes only and is exposed to clients.)

For example, the following is the name form description for the `uddiBusinessEntityNameForm` name form defined in RFC 4403 (`http://www.ietf.org/rfc/rfc4403.txt`):

```
( 1.3.6.1.1.10.15.1 NAME 'uddiBusinessEntityNameForm'
OC uddiBusinessEntity MUST ( uddiBusinessKey ) X-ORIGIN 'RFC 4403' )
```

In this case, the numeric OID is `1.3.6.1.1.10.15.1` and the human-readable name is `uddiBusinessEntityNameForm`. Entries with the `uddiBusinessEntity` structural object class are required to use `uddiBusinessKey` as their only RDN attribute type. There is no description, nor

are there any other attribute types that can optionally be included in the associated entries. The name form is not marked `OBSOLETE`.

10.6 Overview of DIT Content Rules

DIT content rules provide a mechanism for defining the content that can appear in an entry. At most one DIT content rule can be associated with an entry based on its structural object class. If such a rule exists for an entry, then it works with the object classes contained in that entry to define which attribute types must, may, and must not be present in the entry, as well as which auxiliary classes that it may include.

The following sections describe DIT content rules:

- Understanding DIT Content Rule Description Format
- About DIT Content Rule Implementation

10.6.1 Understanding DIT Content Rule Description Format

Learn about the DIT content rule description format and the various elements included in it.

RFC 4512 (`http://www.ietf.org/rfc/rfc4512.txt`), section 4.1.6 describes the DIT content rule description format, as shown here:

```
DITContentRuleDescription = LPAREN WSP
numericoid                ; object identifier
[ SP "NAME" SP qdescrs ]  ; short names (descriptors)
[ SP "DESC" SP qdstring ] ; description
[ SP "OBSOLETE" ]         ; not active
[ SP "AUX" SP oids ]      ; auxiliary object classes
[ SP "MUST" SP oids ]     ; attribute types
[ SP "MAY" SP oids ]      ; attribute types
[ SP "NOT" SP oids ]      ; attribute types
extensions WSP RPAREN     ; extensions
```

The DIT content rule description includes these elements:

- numericoid

  The numeric OID of the structural object class with which the DIT content rule is associated. Although the specification requires a numeric OID, this `numericoid` should match the OID specified for the associated object class, so if the object class OID was non-numeric, then this OID should be as well.

- NAME

  An optional set of human-readable names used to refer to the DIT content rule. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If a description is provided, then it should be enclosed in single quotation marks.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the DIT content rule is active. If a DIT content rule is marked as `OBSOLETE`, then it should not be in effect within Oracle Unified Directory.

- AUX

  An optional list of auxiliary object classes that can be present in entries with the associated structural class. If no values are provided, then such entries are not allowed to have any auxiliary object classes. Values should be specified as one or more of the names or OIDs of the allowed auxiliary classes. If multiple auxiliary classes are allowed, then separate them by spaces and dollar signs, and enclose the entire set of names in parentheses.

- MUST

  An optional list of attribute types that are required to be present in entries with the associated structural class. This is in addition to the attribute types required by the object classes included in the entry, and these additional attribute types do not need to be allowed by any of those object classes. Values should be specified as one or more of the names or OIDs of the required attribute types. If multiple attribute types are required, then separate them by spaces and dollar signs, and enclose the entire set of required attribute types in parentheses.

- MAY

  An optional list of attribute types that can optionally be present in entries with the associated structural class. This is in addition to the attribute types allowed by the object classes included in the entry. Values should be specified as one or more of the names or OIDs of the optional attribute types. If there are multiple optional attribute types, separate them by spaces and dollar signs and enclose the entire set of optional attribute types in parentheses.

- NOT

  An optional list of attribute types that are prohibited from being present in entries with the associated structural class. This list cannot include any attribute types that are required by the structural class or any of the allowed auxiliary classes, but it can be used to prevent the inclusion of attribute types that would otherwise be allowed by one of those object classes. Values should be specified as one or more of the names or OIDs of the prohibited attribute types. If multiple types are prohibited, then separate them by spaces and dollar signs, and enclose the entire set of prohibited attribute types in parentheses.

- extensions

  An optional set of extensions for the DIT content rule.Oracle Unified Directory currently uses the following extensions for DIT content rules:

  - `X-ORIGIN` — Provides information about where the DIT content rule is defined (for example, whether it came from a particular RFC or Internet Draft, or whether it is defined within the project)
  - `X-SCHEMA-FILE` — Indicates which schema file contains the DIT content rule definition (This extension is generally used for internal purposes only and is exposed to clients.)

The following provides an example of a DIT content rule description:

```
( 2.16.840.1.113730.3.2.2 NAME 'inetOrgPersonContentRule'
AUX ( posixAccount $ shadowAccount $ authPasswordObject )
MUST uid )
```

In this case, the numeric OID is `2.16.840.1.113730.3.2.2`, which is the OID for the `inetOrgPerson` structural object class. It has a human-readable name of `inetOrgPersonContentRule` and no description. It allows entries containing the `inetOrgPerson` object class to also contain the `posixAccount`, `shadowAccount`, and `authPasswordObject` auxiliary classes, and those entries must contain the `uid` attribute type. It is not marked `OBSOLETE`, and it does not define any additional optional or prohibited attribute types, nor does it include any extensions.

10.6.2 About DIT Content Rule Implementation

The mechanism used to handle DIT content rule varies from the LDAPv3 specification. Learn more about the importance of implementing DIT content rule.

The LDAPv3 specification states that if the structural object class used in an entry does not have a corresponding DIT content rule, then that entry is not allowed to contain any auxiliary object classes. Oracle Directory Server Enterprise Edition does not support DIT content rules, hence Oracle Unified Directory does not prevent the use of auxiliary object classes in entries for which there is no corresponding DIT content rule. If it is desirable to prevent the inclusion of auxiliary classes in a given type of entry, then a DIT content rule should be created with no allowed auxiliary classes to cover entries with the appropriate structural object class.

10.7 Understanding DIT Structure Rules

DIT structure rules can be used to define the allowed hierarchical structure of the directory data. In particular, they make it possible to specify what types of entries are allowed to exist as immediate children of entries with a specified structural object class. For example, only entries with the `inetOrgPerson` structural class can be immediate children of entries with an `organizationalUnit` structural object class.

DIT structure rules are themselves hierarchical. Each DIT structure rule is assigned a rule ID, which is an integer value, and is also associated with a name form (which in turn links it to one or more structural object classes). DIT structure rules can also reference one or more superior DIT structure rules, and this provides the mechanism for controlling the data hierarchy. If a DIT structure rule does not specify any superior rules, then entries containing its associated structural object class are allowed to exist at the root of the associated schema. If a DIT structure does specify one or more superior rules, then entries with an associated structural object class are allowed to exist only below entries containing the structural object class of one of those superior rules.

The following sections describe DIT structure rules:

- Understanding DIT Structure Rule Description Format
- Understanding DIT Structure Rules and Multiple Schemas

10.7.1 Understanding DIT Structure Rule Description Format

Learn about the DIT structure rule description format and the various elements included in it.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.7.1 describes the DIT structure rule description format, as shown here:

```
DITStructureRuleDescription = LPAREN WSP
ruleid                      ; rule identifier
[ SP "NAME" SP qdescrs ]    ; short names (descriptors)
[ SP "DESC" SP qdstring ]   ; description
[ SP "OBSOLETE" ]           ; not active
SP "FORM" SP oid            ; NameForm
[ SP "SUP" ruleids ]        ; superior rules
extensions WSP RPAREN       ; extensions
ruleids = ruleid / ( LPAREN WSP ruleidlist WSP RPAREN )
ruleidlist = ruleid *( SP ruleid )
ruleid = number
```

The DIT structure rule description includes these elements:

- ruleid

  The integer rule ID assigned to the DIT structure rule. It must be unique among all other DIT structure rules in the schema.

- NAME

  An optional set of human-readable names that can be used to refer to the DIT structure rule. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If a description is provided, then it should be enclosed in single quotes.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the DIT structure rule is active. If it is marked `OBSOLETE`, then it should not be taken into account when entries are created or moved.

- FORM

  The name or OID of the name form with which the DIT structure rule is associated. As mentioned in Understanding DIT Structure Rules, the name form associates the DIT structure rule with a structural object class.

- SUP

  An optional set of superior rule IDs for the DIT structure rule. If there are multiple superior rule IDs, then separate them by spaces, and enclose the entire set of superior rule IDs in parentheses. It is permissible for multiple DIT structure rules to use overlapping sets of superior rule IDs.

- extensions

  An optional set of extensions for the DIT structure rule.Oracle Unified Directory currently uses the following extensions for DIT structure rules:

  - `X-ORIGIN` — Provides information about where the DIT structure rule is defined (for example, whether it came from a particular RFC or Internet Draft, or whether it is defined within the project)
  - `X-SCHEMA-FILE` — Indicates which schema file contains the DIT structure rule definition (This extension is generally used for internal purposes only and is exposed to clients.)

The following example is the DIT structure rule definition for the `uddiContactStructureRule` DIT structure rule:

```
dITStructureRule:
( 2 NAME 'uddiContactStructureRule' FORM uddiContactNameForm SUP ( 1 )
X-ORIGIN 'RFC 4403' )
```

In this case, the rule ID is `2`, and the human-readable name is `uddiContactStructureRule`. It is associated with the `uddiContactNameForm` name form (which in turn links it to the `uddiContact` object class), and it has a superior rule ID of `1`. It was defined in RFC 4403 (`http://www.ietf.org/rfc/rfc4403.txt`). It does not have a description, nor is it marked `OBSOLETE`.

10.7.2 Understanding DIT Structure Rules and Multiple Schemas

DIT structure rules can provide a mechanism for placing constraints on Oracle Unified Directory hierarchy, but to maximize their utility, it may be necessary to use them with support for multiple schemas.

For example, consider a directory with a naming context of `dc=example,dc=com`, below which are two branches: `ou=People,dc=example,dc=com` and `ou=Groups,dc=example,dc=com`. If you want to allow only `inetOrgPerson` entries below the `ou=People` branch and only `groupOfNames` entries below the `ou=Groups` branch, then that can be fully accomplished only if there are different schemas that govern the `ou=People` and `ou=Groups` branches.

If there were a single schema governing the entire directory server, then you can imagine that it would have four DIT structure rules:

- `dITStructureRule: (11 NAME 'domainStructureRule' FORM domainNameForm)`
- `dITStructureRule: (12 NAME 'organizationalUnitStructureRule' FORM organizationalUnitNameForm SUP 11)`
- `dITStructureRule: (13 NAME 'inetOrgPersonStructureRule' FORM inetOrgPersonNameForm SUP 12)`
- `dITStructureRule: (14 NAME 'groupOfNamesStructureRule' FORM groupOfNamesNameForm SUP 12)`

This set of DIT structure rules would allow the structure described above, but it would also allow the creation of group entries below the `ou=People` branch and the creation of user entries below the `ou=Groups` branch. The only way to prevent that using DIT structure rules would be to define separate schemas for the `ou=People` and `ou=Groups` branches and define only the `inetOrgPersonStructureRule` rule in the schema for the `ou=People` branch, and only define the `groupOfNamesStructureRule` rule in the schema for the `ou=Groups` branch.

10.8 Understanding Matching Rule Uses

You can use matching rule uses to specify which attribute types can be used with a given matching rule when processing a search request with an extensible match filter component. If that extensible match component includes both an attribute type and a matching rule ID, then Oracle Unified Directory checks to see if there is a matching rule use for the associated matching rule, and if there is, it ensures that it allows the specified attribute type to be used with that matching rule.

RFC 4512 (http://www.ietf.org/rfc/rfc4512.txt), section 4.1.4 describes the matching rule use description format, as shown here:

```
MatchingRuleUseDescription = LPAREN WSP
numericoid                 ; object identifier
[ SP "NAME" SP qdescrs ]   ; short names (descriptors)
[ SP "DESC" SP qdstring ]  ; description
[ SP "OBSOLETE" ]          ; not active
SP "APPLIES" SP oids       ; attribute types
extensions WSP RPAREN      ; extensions
```

The matching rule use description includes these elements:

- numericoid

The numeric OID of the matching rule with which the matching rule use is associated. There can be only one matching rule use associated with a given matching rule.

- NAME

  An optional set of human-readable names that may be used to refer to the matching rule use. If there is a single name, then it should be enclosed in single quotes. If there are multiple names, then they should each be enclosed in single quotes and separated by spaces, and the entire set of names should be enclosed in parentheses.

- DESC

  An optional human-readable description. If there is a description, then it should be enclosed in single quotes.

- OBSOLETE

  An optional `OBSOLETE` flag that can be used to indicate whether the matching rule use is active. If it is marked `OBSOLETE`, then it should not be taken into account when determining whether to allow an extensible match filter.

- APPLIES

  A set of one or more attribute types that can be used with the associated matching rule. If there is an associated attribute type, then its name or OID can be used. If there are multiple attribute types, then separate them by spaces and dollar signs, and enclose the entire set of associated attribute types in parentheses.

- extensions

  An optional set of extensions for the matching rule use.Oracle Unified Directory currently uses the following extensions for matching rule uses:

  - `X-ORIGIN` — Provides information about where the matching rule is defined (for example, whether it came from a particular RFC or Internet Draft, or whether it is defined within the project)
  - `X-SCHEMA-FILE` — Indicates which schema file contains the matching rule definition (This extension is generally used for internal purposes only and is exposed to clients.)

The following example shows a matching rule use description:

```
( 1.3.6.1.4.1.26027.1.999.10 NAME 'testAddMRUSuccessful' APPLIES cn )
```

In this case, the numeric OID is `1.3.6.1.4.1.26027.1.999.10`, the single human-readable name is `testAddMRUSuccessful`, and it can be used with the `cn` attribute. It does not have a description, it is not marked `OBSOLETE`, and it does not have any extensions.

11

Understanding Root Users and the Privilege Subsystem

Most LDAP directory servers typically have a single superuser, which is much like the root account in traditional UNIX systems. This account can bypass access controls and other restrictions that might be enforced for regular users. In Oracle Unified Directory you can define multiple root users, and a privilege subsystem that makes it possible to control capabilities at a more fine-grained level.
The following topics describe root user accounts and the privilege subsystem:

11.1 About Root User Accounts

Root user accounts are defined below the `cn=Root DNs,cn=config` branch in the server configuration. Each root account is defined as a regular user entry, except that it includes the `ds-cfg-root-dn-user` auxiliary object class.

A root user entry can also have one or more values for the `ds-cfg-alternate-bind-dn` attribute. This attribute specifies alternate DNs that can be used to authenticate as that user (for example, so you can bind as `cn=Directory Manager` instead of having to use `cn=Directory Manager,cn=Root DNs,cn=config`, which is the actual entry DN).

The ability to define multiple root users, each in its own entry, provides the following advantages:

- Each administrator that needs root access to the directory server can have their own account with their own credentials. This makes it easier to keep an audit trail of who does what in the directory server than if all of the administrators shared a single root account.
- Because each root user account has its own set of credentials, the credentials for one root user can be changed without impacting any of the other root users. It is not necessary to coordinate root password changes among all of the administrators because each of them has their own account. If an administrator leaves, that account can simply be deactivated or removed.
- Because each root user has its own entry, and you can put whatever attributes and object classes you want into that entry (if it also has the `ds-cfg-root-dn-user` auxiliary object class), root users are capable of using strong authentication like the EXTERNAL or GSSAPI SASL mechanisms.
- Root users are subject to password policy enforcement. This means that you can force root users to change their passwords on a regular basis, ensure that they are only allowed to authenticate or change their passwords using secure mechanisms, and ensure that they choose strong passwords. You can also use custom password policies for root users, so that they are subject to different sets of password policy requirements than other users in the directory.

- You can define different resource limits for root users than for regular users. Because each root account has its own entry, operational attributes like `ds-rlim-size-limit`, `ds-rlim-time-limit`, and `ds-rlim-lookthrough-limit` work for root users just as they do with regular user accounts.
- Only root users can bind to the administration port because administrative binds are resolved with root `dns` from `cn=config`. To create a root `dn`, see Creating a Root User.

11.2 Understanding Privilege Subsystem

Root user accounts in traditional directories are special because they can bypass access controls and other restrictions, and there are some kinds of operations that only root users can perform.

This is much like the concept of root users in traditional UNIX operating systems. However, there might be cases in which a regular user needs to do something that only a root user can do. If users are given root access, they are given far more power than they actually need to do their job, and system administrators have to hope that they use this power responsibly and do not intentionally or unintentionally impact some other part of the system. Alternately, the user might not be given root access and either not be able to perform a vital function or have to rely on one of the system administrators to perform the task.

Solaris 10 and onward address this problem in UNIX systems by creating a privilege subsystem (also called "process rights management"). The engineers developing Solaris realized that it is dangerous and undesirable to be forced to give someone root access just to perform one specific task. For example, just because a user may need to start a process that listens on a port below 1024 does not mean that they should also be able to bypass filesystem permissions, change network interface settings, or mount and unmount file systems. With the privilege subsystem in Solaris 10, it is possible to give a user just the specific capability that they need, for example, the ability to bind to privileged ports, without giving them full root access. Similarly, it is possible to take away privileges that might otherwise be available. For example, an account that is only used to run a specific daemon does not need to be able to see processes owned by other users on the system.

> **Note:**
>
> Administrators should consider Oracle Privileged Account Management system to achieve the best security level.

Oracle Unified Directory also has a privilege subsystem that defines distinct capabilities that users might need and makes it possible to give them just the level of access that they require. Regular users can be granted privileges that they would not otherwise have, certain privileges can be taken away from root users. The set of privileges currently defined in the directory server are described below:

Privileges	Description
bypass-acl	Allows the user to bypass access control evaluation
modify-acl	Allows the user to make changes to the access controls defined in the server
config-read	Allows the user to have read access to the server configuration
config-write	Allows the user to have write access to the server configuration
jmx-read	Allows the user to read JMX attribute values

Privileges	Description
jmx-write	Allows the user to update JMX attribute values
jmx-notify	* Allows the user to subscribe to JMX notifications
ldif-import	Allows the user to request the LDIF import task
ldif-export	Allows the user to request the LDIF export task
backend-backup	Allows the user to request the back end backup task
backend-restore	Allows the user to request the back end restore task
server-shutdown	Allows the user to request the server shutdown task
server-restart	Allows the user to request the server restart task
proxied-auth	Allows the user to use the proxied authorization control or request an alternate SASL authorization ID
disconnect-client	Allows the user to terminate arbitrary client connections
cancel-request	* Allows the user to cancel arbitrary client requests
unindexed-search	Allows the user to request unindexed search operations
password-reset	Allows the user to reset the passwords for other users
update-schema	Allows the user to update the server schema
privilege-change	Allows the user to change the set of privileges assigned to a user, or to change the set of default root privileges

Currently, the privileges marked with an asterisk (*) are not yet implemented in the server and therefore have no effect.

The privilege subsystem is largely independent of the access control subsystem. Unless the user also has the `bypass-acl` privilege, operations might still be subject to access control checking. For example, if a user has the `config-read` privilege, that user can see only those parts of the configuration that are allowed by access control. As a rule, whenever an operation is covered by both the privilege subsystem and access control, both mechanisms must allow that operation.

11.3 Assigning Privileges to Normal Users

Privileges are assigned by adding the `ds-privilege-name` operational attribute to the user's entry. By default, normal users are not granted any of the privileges. Therefore, if a user should be allowed to perform any of the associated operations, they must be granted the appropriate privileges.

See Understanding Privilege Subsystem for more information on the set of privileges currently defined in the directory server.

> **Note:**
>
> Adding a privilege with a value such as `modify-acl` is not sufficient for granting a user the right to add, replace, or delete an ACI. Appropriate access control for the user to modify the ACI for another entry is also required. See Understanding the Syntax of Access Control Instructions for more information.

`ds-privilege-name` is a multivalued attribute, and if a user is to be given multiple privileges, then a separate value should be used for each one. When the virtual attribute subsystem is in place, it should also be possible to grant privileges to groups of users automatically by making `ds-privilege-name` a virtual attribute in those user entries.

As an example, the following modification can be used to add the `proxied-auth` privilege to the user `cn=Proxy User,dc=example,dc=com`:

```
dn: cn=Proxy User,dc=example,dc=com
changetype: modify
add: ds-privilege-name
ds-privilege-name: proxied-auth
```

> **Note:**
>
> If you want the modifications of the privileges of a user to take effect on an open connection after the first bind, then you must set the `maintain-authenticated-users` flag to `true`. By default, it is set to `false`.
>
> For an open connection, which is bound with a determined `authDN`, importing that entry with `dn: authDN` using `import-ldif` command does not modify the properties (access rights, privileges, and so on) of that `authDN` in those already established connections. The new properties for the `authDN` as a result of `import-ldif` are effective only for new binds as `authDN`. In this scenario, setting `maintain-authenticated-users:true` does not help.

11.4 Assigning Privileges to Root Users

Root users too are not granted with any of the privileges, by default. Therefore, if a root user should be allowed to perform any of the associated operations, they must be granted the appropriate privileges.

The following topics describe how to assign privileges to root users and what are the distinguishing characteristics:

- About Privileges Assigned to Root Users
- Modifying Privileges Assigned to Root Users

11.4.1 About Privileges Assigned to Root Users

With the introduction of the privilege subsystem, the primary distinguishing characteristics of root users that separate them from other accounts in the server are that they exist in the configuration rather than in the user data. Moreover, because they are root users, they automatically inherit a certain set of privileges.

The set of privileges automatically granted to root users is defined in the `ds-cfg-default-root-privilege-name` attribute of the `cn=Root DNs,cn=config` entry. By default, root users are automatically granted the following privileges:

- `bypass-acl`
- `modify-acl`
- `config-read`

- config-write
- ldif-import
- ldif-export
- backend-backup
- backend-restore
- server-shutdown
- server-restart
- disconnect-client
- cancel-request
- unindexed-search
- password-reset
- update-schema
- privilege-change

11.4.2 Modifying Privileges Assigned to Root Users

If you want to alter the set of privileges that are automatically assigned to root users, then you may do so by editing the `ds-cfg-default-root-privilege-name` attribute. Further, if you want to have a different set of privileges for a specific root user, then you can accomplish that using the `ds-privilege-name` attribute in that root user's entry, just like for a normal user.

For example, the following modification may be used to give a specific root user (in this case `cn=Test Root User,cn=Root DNs,cn=config`) the ability to use proxied authorization while removing the ability to change user privileges or access the configuration. (The minus sign before the privilege indicates that it is being removed rather than granted.):

```
dn: cn=Test Root User,cn=Root DNs,cn=config
changetype: modify
add: ds-privilege-name
ds-privilege-name: proxied-auth
ds-privilege-name: -config-read
ds-privilege-name: -config-write
```

In this case, the `cn=Test Root User,cn=Root DNs,cn=config` user inherits all privileges automatically granted to root users with the exception of the `config-read` and `config-write` privileges and is also given the `proxied-auth` privilege.

12

Understanding the Proxy, Distribution, and Virtualization Functionality

Understand about the conceptual overview of the Oracle Unified Directory functionality that enables you to use a proxy server for various types of deployments.
This functionality includes configurable workflow elements and an extensible plug-in API that you can use to work with data residing on remote and distributed data sources or servers.

This chapter includes the following sections:

- Accessing Remote Data Sources
- Overview of Load Balancing Using the Proxy
- Overview of Data Distribution Using the Proxy
- Understanding Data Integration Using the Proxy
- Understanding Virtualization
- Understanding the Global Index Catalog
- Understanding the Transformation Framework

> **Note:**
>
> Before reading this chapter, review Introduction to Oracle Unified Directory and Understanding Deployments Using the Proxy Server for a better understanding of the concepts described here.
>
> For more information about configuring the features and functionality described in this chapter, see the chapters in Configuring Proxy, Distribution, and Virtualization Functionality

12.1 Accessing Remote Data Sources

Remote data is accessed either in a relational database management system (RDBMS) such as an Oracle Database or a remote LDAP directory server.

The following topics describe how to access remote data:

- Enabling LDAP Clients to Access Identity Data Stored in an RDBMS
- Understanding How to Enable Communication with a Remote LDAP Server

For configuration information, see Configuring Access to Remote Data Sources .

12.1.1 Enabling LDAP Clients to Access Identity Data Stored in an RDBMS

The RDBMS workflow element enables LDAP clients to access identity data stored in a relational database management system (RDBMS) using the LDAP protocol.

The topics in this section include:

- Understanding How to Use an RDBMS Workflow Element
- About RDBMS Workflow Element Features
- Caching RDBMS Workflow Element
- Configuring RDBMS Workflow Element

For information about configuring an RDBMS workflow element and its supporting components, see Configuring Access to Identity Data Stored in an RDBMS.

12.1.1.1 Understanding How to Use an RDBMS Workflow Element

The RDBMS workflow element allows you to create a bridge between Oracle Unified Directory LDAP clients and an RDBMS such as an Oracle Database. A deployment can use an RDBMS workflow element implementation to meet the following requirements:

- The deployment stores some identity data in an LDAP directory server, but it also has additional data stored in an RDBMS. LDAP clients want to integrate data from both sources into aggregate virtual views.
- LDAP clients want to use the LDAP protocol to read and write the identity data stored in both the LDAP directory server and the RDBMS. These clients do not want to use SQL queries and commands to access the RDBMS data.

12.1.1.2 About RDBMS Workflow Element Features

An RDBMS workflow element implementation supports the following features:

- You can configure a connection to most RDBMS databases that support JDBC. For a list of supported databases, check the Oracle Unified Directory 14c Release (14.1.2.1.0) Certification Matrix.
- You can map LDAP object classes and attributes to SQL tables and columns in the RDBMS to create virtual views of the RDBMS data. You are not required to make any modifications to the RDBMS.
- You can use the following LDAP operations. These operations are translated to the equivalent SQL queries to access data stored in the RDBMS:
  - `BIND`
  - `ADD`
  - `DELETE`
  - `MODIFY`
  - `MODIFYDN`
  - `SEARCH`

> **Note:**
>
> In the current release, the RDBMS workflow element does not support LDAP write operations (add, modify, or delete) when entries are built from multiple SQL tables.

- You can control access to the virtual views of the RDBMS data using an access control group and virtual ACIs based on LDAP client identities.

12.1.1.3 Caching RDBMS Workflow Element

The RDBMS workflow element uses eclipselink to maintain an in-memory cache containing data already accessed from the RDBMS. By default, this cache is enabled.

The default (and recommended) `caching-scheme` is `soft-weak`. This scheme holds soft references to the database entries and enables optimal caching of objects while still allowing the JVM to garbage collect them if memory is low. This scheme also maintains a most frequently used subcache containing soft references to the objects, which allows the objects to be garbage collected except for a fixed number of the most recently used objects.

To specify a different caching scheme for the RDBMS workflow element, use `dsconfig`. For example:

```
$ dsconfig set-workflow-element \
--element-name ORCL1 \
--set caching-scheme:full
```

When data entries targeted by the proxy are modified by external means (for example, by an application or a user directly accessing the database using an SQL statement), the changes might not be reflected in the RDBMS workflow element. If you require strong data consistency, change the caching scheme or disable caching altogether by setting the `caching-scheme` property in the RDBMS workflow element.

For example, to disable caching, set the `caching-scheme` property to `none`:

```
$ dsconfig set-workflow-element \
--element-name ORCL1 \
--set caching-scheme:none
```

For more information about the RDBMS workflow element caching properties, including the caching schemes you can configure, see the *Oracle Fusion Middleware Configuration Reference for Oracle Unified Directory*.

12.1.1.4 Configuring RDBMS Workflow Element

To implement an RDBMS workflow element, you need to configure the following components:

- Configuring Oracle Unified Directory Proxy Server
- Installing JDBC Driver JAR File
- Creating RDBMS Workflow Element and Supporting Components
- Configuring Access Control Group and Virtual ACIs

For information about configuring the components, see Configuring Access to Identity Data Stored in an RDBMS.

12.1.1.4.1 Configuring Oracle Unified Directory Proxy Server

The RDBMS workflow element requires an Oracle Unified Directory proxy server as the interface between the LDAP clients and the RDBMS. The proxy server uses the following elements to communicate with the RDBMS:

- An RDBMS extension manages the connectivity with the remote server through JDBC by periodically checking the response from the remote peer and providing valid connections maintained by the connection pool.
- An RDBMS workflow element retrieves the connections from the RDBMS extension element, performs mapping between LDAP entries and SQL tables, and executes operations received from the LDAP clients.

To create a proxy server, you run the `oud-proxy-setup` or `oud-proxy-setup.bat` script.

12.1.1.4.2 Installing JDBC Driver JAR File

An RDBMS workflow element implementation relies on the JDBC standard to communicate with the RDBMS. If you are using Oracle Database, JDBC driver is already included and no action is needed. However, if you are using a non Oracle Database, you must install the JDBC driver JAR file that corresponds to the RDBMS you are using.

12.1.1.4.3 Creating RDBMS Workflow Element and Supporting Components

Communication with the RDBMS requires the RDBMS workflow element and its components. To create and configure these components, you perform the following tasks:

1. Create an RDBMS extension, RDBMS workflow element, and a workflow associated with the RDBMS workflow element.
2. Assign the workflow associated with the RDBMS workflow element to a network group.
3. Configure LDAP-SQL mappings for the LDAP attributes and object classes that correspond to the SQL tables and columns you want to access in the RDBMS.

12.1.1.4.4 Configuring Access Control Group and Virtual ACIs

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

Access control to the virtual data from the RDBMS is configured using an access control group with virtual ACIs based on the LDAP client identities. Virtual ACIs are created and stored on the Oracle Unified Directory proxy instance.

To configure access control to the virtual data, you perform the following tasks:

1. Create an access control group for the workflow associated with the RDBMS workflow element.
2. Create virtual ACIs based on the LDAP client identities and add these virtual ACIs to the access control group created in Step 1.

> **Note:**
>
> Your access control strategy for the virtual data from the RDBMS depends on your corporate policies, so you must create virtual ACIs to follow those policies.

For more information, see Understanding Virtual Access Control Instructions.

12.1.2 Understanding How to Enable Communication with a Remote LDAP Server

You can enable communication with a remote LDAP server by using LDAP Server Extension and Proxy LDAP Workflow elements.

To enable communication between a proxy instance and a remote LDAP server:

- **LDAP Server Extension:** This element manages connectivity with a remote server by periodically checking the response from the remote peer and providing valid connections maintained by the connection pool.
- **Proxy LDAP Workflow Element:** This element retrieves connections from the LDAP server extension element and executes operations received from the user as defined in the configured mode.

> **Note:**
>
> - For information about configuring an LDAP server extension, see Configuring LDAP Server Extensions.
> - For information about configuring a Proxy LDAP workflow element, see Configuring Proxy LDAP Workflow Elements.

12.2 Overview of Load Balancing Using the Proxy

You can use the proxy to load balance requests across multiple data sources or replicated LDAP servers.

> **Note:**
>
> For information about how to configure load balancing, see Configuring Load Balancing Using the Proxy .

This section contains the following topics:

- Understanding Load Balancing Using the Proxy
- Understanding Failover Load Balancing
- Understanding Optimal Load Balancing

- Understanding Proportional Load Balancing
- Understanding Saturation Load Balancing
- Understanding Search Filter Load Balancing

12.2.1 Understanding Load Balancing Using the Proxy

In a load balancing deployment, the requests are routed to one of the data sources based on the *load balancing algorithm* set. There are five load balancing algorithms: Failover, Optimal, Proportional, Saturation and Search Filter.

You can choose one of the following load balancing algorithms:

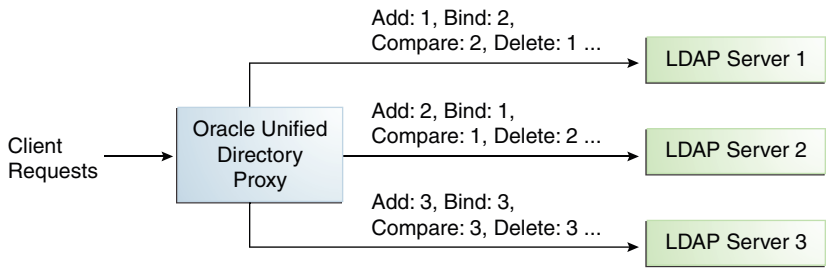
- **Failover**. Several remote LDAP server handle requests, based on the priority configured on a server, for a given operation type. When there is a failure, requests are sent to the server with the next highest priority for that operation type.
- **Optimal**. There is no priority between the different remote LDAP servers. The LDAP server with the lowest saturation level is the one that handles the requests. The saturation level of the remote LDAP servers is regularly reevaluated, to ensure that the best route is chosen.
- **Proportional**. All the remote LDAP servers handle requests, based on the proportions (weight) set.
- **Saturation**. There is one main LDAP server that handles all requests, until the saturation limit is reached.
- **Search Filter**. Several LDAP servers are deployed, and handle requests based on certain attributes in the request search filter.

12.2.2 Understanding Failover Load Balancing

In a load balancing with failover algorithm, the proxy routes requests to the remote LDAP server or data center with the highest priority for a given operation type, for example for Add operations. The proxy continues to send requests to the priority route until the remote LDAP server goes down. This may be caused by a network cut, a hardware failure, a software failure or some other problem. At failover, the proxy routes incoming requests to the server with the second highest priority for that specific operation type.

Figure 12-1 illustrates a failover load balancing configuration. In this example, there are three routes, each with a unique priority per operation type. All Add operations are treated by Server 1, since it has the highest priority, that is `priority=1`, while Bind operations are handled by Server 2. If Server 1 goes down, the Add requests are sent to the server with the second highest priority, that is, Server 2.

Figure 12-1 Failover Load Balancing Example



By default, the proxy does not immediately reroute requests to a server that has gone down, once it is running again. For example, if Server 1 goes down, the Add requests are sent to Server 2. Even when Server 1 is up again, Server 2 continues to handle incoming Add requests. However, if Server 2 goes down, and Server 1 is up again, Server 1 will now receive incoming requests. This default behavior can be changed with the `switch-back` flag. For information about configuring the `switch-back` flag, see Setting the `switch-back` Flag.

For failover to work effectively, the monitoring check interval must be set to be low enough so that the failover happens inside a time interval that suits your business needs. For details about setting the monitoring check interval, see Modifying the LDAP Data Source Monitoring Connection Properties.

12.2.3 Understanding Optimal Load Balancing

Understand optimal load balancing concepts and learn how to determine saturation level.

The following sections provide an overview of optimal load balancing concepts and how to determine saturation level:

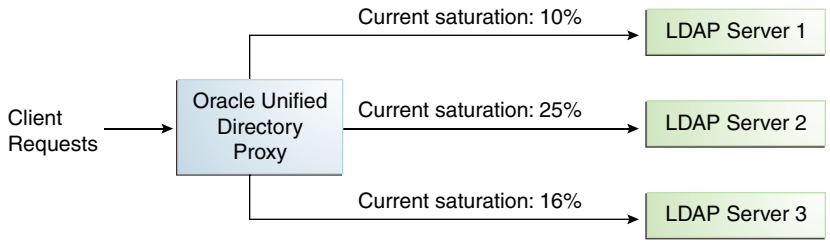
- Overview of Optimal Load Balancing
- Determining Saturation Levels

12.2.3.1 Overview of Optimal Load Balancing

With the optimal load balancing algorithm, the proxy sends requests to the route with the lowest saturation level. The proxy continues to send requests to this route until the saturation level of the remote LDAP server on that route passes the saturation level of the other remote LDAP servers in the deployment. The saturation level is represented as a percentage.

When the saturation level of a route changes, the load balancing algorithm re-evaluates the best route and if required, selects another route as the active one. The route with the lowest saturation level is always chosen as the optimal route. In the configuration illustrated by Figure 12-5, Server 1 has the lowest saturation level and will handle all the requests until its saturation level rises above the saturation level of the other servers. If one of the servers goes down, its saturation level is considered as 100%.

Figure 12-2 Optimal Load Balancing Example



You can configure the saturation precision, to set the difference of saturation between two servers before the route changes to the server with the lowest saturation level. By default, the saturation precision is set to 5. However, if you find that the algorithm is switching between servers too often, you can set the saturation precision to 10, for example. The saturation precision is set in the LDAP server extension, see Setting the Saturation Precision for the Optimal or Saturation Algorithm.

12.2.3.2 Determining Saturation Levels

The saturation level is a ratio between the number of connections in use in the connection pool and its configured maximum size. The connection pool maximum size is an advanced parameter of the LDAP server extension object.

If the number of connections in use is lower than the maximum pool size divided by 2, then the saturation is `0`. This implies that the pool is not saturated.

When more than half of the connections are in use, the saturation level is calculated as follows:

100 * (1 - available connections/(max pool size/2))

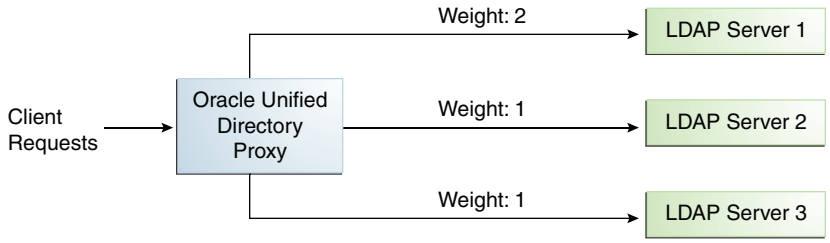
This implies that the saturation level is `100` when all the connections are in use.

12.2.4 Understanding Proportional Load Balancing

With the proportional load balancing algorithm, the proxy forwards requests across multiple routes to remote LDAP servers or data sources, based on the proportions set. The proportion of requests handled by a route is identified by the weight that you set for each route in your configuration. The weight is represented as an integer value.

When you configure load balancing, you must indicate the proportion of requests handled by each LDAP server. In the example in Figure 12-3, Server 1 handles twice as many connections as Server 2, since the weight is set with a proportion of 2:1. Server 2 and Server 3 handle the same amount of requests (1:1).

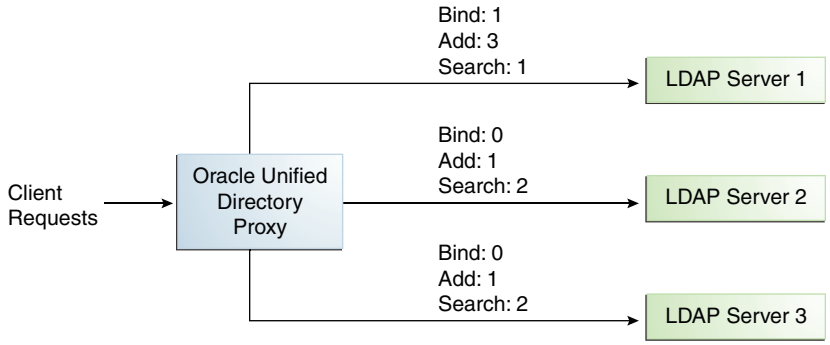
Figure 12-3 Proportional Load Balancing Example



You can configure a specific weight for each type of client operation, as illustrated in Figure 12-4. For example, in you want Server 1 to handle all the Bind operations, this is possible. To do so, set the weight of bind to `1` (or higher) for Server 1, and to `0` for Server 2 and Server 3.

In the example illustrated in Figure 12-4, Server 1 will handle three times as many Add requests as Server 2 and Server 3. However, Server 1 will handle only one half the Search requests handled by Server 2, and Server 3. Server 2 and Server 3 will handle the same amount of Add and Search requests, but will not handle Bind requests.

Figure 12-4 Proportional Load Balancing with Request Specific Management



If you do not modify the weights of operations other than Bind, Add, and Search, as illustrated in Figure 12-4, the servers will share the same load for all other operations (for example for Delete operations).

For more information on configuring the load balancing weights of routes when using proportional load balancing, see Modifying Load Balancing Properties.

12.2.5 Understanding Saturation Load Balancing

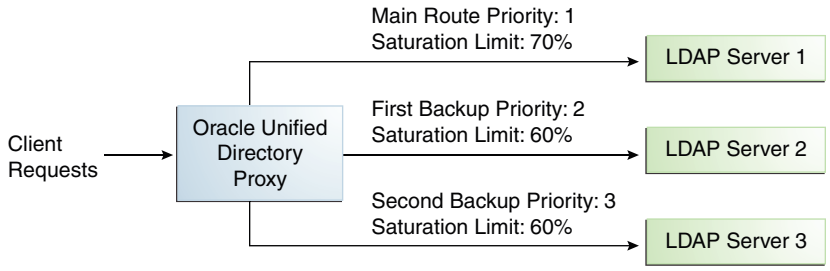
Understand, with an example, about Saturation Load Balancing and what happens with saturation load balancing algorithm. With the saturation load balancing algorithm, the proxy sends requests to a chosen priority route. The proxy continues to send requests to the priority route until the remote LDAP server on that route passes the saturation threshold set. The saturation threshold is represented as a percentage.

For example, if you want a remote LDAP server to manage all incoming requests, set it as priority 1. If you want that same remote LDAP server to stop handling requests when its saturation index reaches 70%, set the saturation threshold to 70%, as illustrated in Figure 12-5. In this way, the server handles all incoming requests until it becomes 70% saturated. The proxy then sends all new requests to the remote LDAP server to Server 2, since it has the next highest priority. Server 2 will continue to handle requests until it reaches its own saturation threshold, or until Server 1 is no longer saturated.

In other words, if Server 1 reaches 70% saturation, the proxy directs the requests to Server 2. If Server 1 is still at 70%, and Server 2 reaches 60%, the proxy directs the new requests to Server 3.

However, if while Server 2 is handling requests, the saturation level of Server 1 drops to 55%, the proxy will direct all new requests to Server 1, even if Server 2 has not reached its saturation threshold.

Figure 12-5 Saturation Load Balancing Example



If all routes have reached their saturation threshold, the proxy chooses the route with the lowest saturation.

You can set a saturation threshold alert that warns you when a server reaches its saturation limit. For example, if you set a saturation threshold alert to 60%, you will receive a notification when the server reaches this limit, and you can act before the server becomes too degraded.

For more information about how to determine the saturation level, see Determining Saturation Levels.

12.2.6 Understanding Search Filter Load Balancing

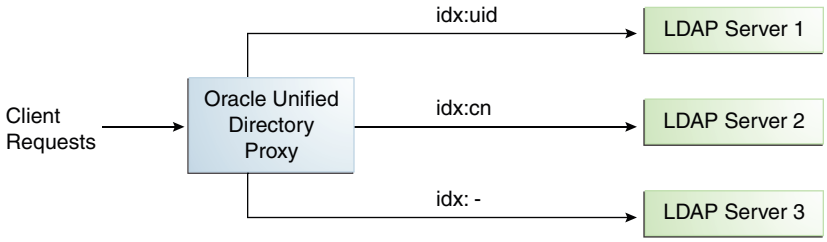
Understand about search filter load balancing with the following example.

With the search filter load balancing algorithm, the proxy routes search requests to LDAP servers based on the presence of certain attributes defined in the request search filter.

The topology consists of several LDAP servers that are accessible through the proxy. All the LDAP servers contain similar data, but each server is optimized based on attributes defined in the search filter to provide better performance. You can configure each route with a list of allowed attributes and a list of prohibited attributes. A search request matches a route when the request search filter contains at least one allowed attribute, and none of the prohibited attributes.

The Figure 12-6 illustrates a search filter load balancing algorithm. In this example, there are three LDAP servers and therefore three distinct routes. LDAP server 1 indexes the `uid` attribute, LDAP server 2 indexes the `cn` attribute, and the third LDAP server is a pass-through route.

Figure 12-6 Search Filter Load Balancing



When the proxy receives a search request that contains the `uid` attribute in its search filter, the search request is routed to LDAP server 1 for better performance. Similarly, if the search filter

contains a `cn` attribute, then the search request is routed to LDAP server 2. All other search requests are routed to the pass-through LDAP server 3.

All other requests, such as ADD, DELETE, MODIFY, and so on can be routed to any LDAP server based on the highest priority. Each search filter route is given a priority. This priority determines the order in which the route are evaluated. The highest priority route filter that matches the search filter is selected to process the request. If all the search filter routes have the same priority, then any route can process the request.

12.3 Overview of Data Distribution Using the Proxy

The Oracle Unified Directory distribution feature addresses the challenge of large deployments, such as horizontal scalability, where all the entries cannot be held on a single data source, or LDAP server. Using distribution can also help you scale the number of updates per second.

This section contains the following topics:

- Understanding Data Distribution Using the Proxy
- Understanding Numeric Distribution
- Understanding Lexico Distribution
- Understanding Capacity Distribution
- Understanding DN Pattern Distribution
- Understanding Union Workflow Element

12.3.1 Understanding Data Distribution Using the Proxy

You can understand about data distribution using the proxy. In a distribution deployment, you must first split your data into smaller chunks. To split the data, you can use the `split-ldif` command. These chunks of data are called *partitions*. Typically, each partition is stored on a separate server

> **Note:**
>
> For information about configuring data distribution, see Configuring Distribution Using the Proxy .

The data is split based on one of the following distribution algorithms:

- Numeric. Entries are split into partitions and distributed based on the numeric value of the naming attribute (for example uid).
- Lexico. Entries are split into partitions and distributed based on the alphabetical value of the naming attribute (for example cn).
- Capacity. Entries are added to a partition based on the capacity of each partition. This algorithm is used for Add requests only. All other requests are distributed by the global index catalog or by a broadcast.
- DN pattern. Entries are split into partitions and distributed based on the pattern (value) of the entry DN.

The type of data distribution you choose will depend on how the data in your directory service is organized. Numeric and lexico distribution have a very specific format for distribution. DN pattern can be adapted to match an existing data distribution model.

If a client request (except Add) cannot be linked to one of the distribution partitions, the proxy broadcasts the incoming request to all the partitions, unless a *global index catalog* has been configured.

However, if the request is clearly identified as outside the scope of the distribution, the request is returned with an error indicating that the entry does not exist. For example, if the distribution partitions includes data with uid's from 1-100 (`partition1`) and 100-200 (`partition2`) but you run a search where the base DN is `uid=222,ou=people,dc=example,dc=com`, the proxy will indicate that the entry does not exist.

Moreover, for the numeric and lexico algorithms, it is the *first RDN* after the distribution base DN that is used to treat a request. For example, the following search will return an error, as the uid is not the first RDN after the distribution base DN, for example `ou=people,dc=example,dc=com`.

```
$ ldapsearch -b "uid=1010,o=admin,ou=people,dc=example,dc=com" "objectclass=*"
```

Consider the number of partitions carefully. When defining the number of partitions for your deployment, be aware that you cannot split and redistribute the data into new partitions without downtime. You can, however, add a new partition with data that has entries outside the initial ones.

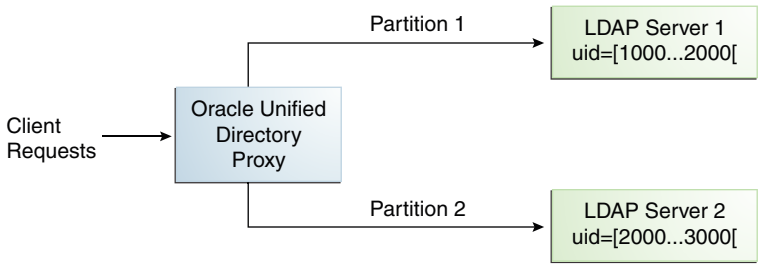
For example, if the initial partitions cover data with uids from 1-100 (`partition1`) and 100-200 (`partition2`), you can later add a `partition3` which includes uids from 200-300. However, you cannot easily split `partition1` and `partition2` so that `partition1` includes uids 1-150 and `partition2` includes uids 150-300, for example. Splitting partitions is essentially like reconfiguring a new distribution deployment.

12.3.2 Understanding Numeric Distribution

Understand numeric distribution with the following example. With a distribution using numeric algorithm, the proxy forwards requests to one of the partitions, based on the numeric value of the first RDN after the distribution base DN in the request. When you set up distribution with numeric algorithm, you split the data of your database into different partitions based on a numerical value of the attribute of your choice, if the attribute represents a numerical string. The proxy then forwards all client requests to the appropriate partition, using the same numeric algorithm.

For example, you could split your data into two partitions based on the uid of the entries, as illustrated in Figure 12-7.

Figure 12-7 Numeric Distribution Example



In this example, a search for an entry with a uid of `1111` is sent to Partition 1, while a search for an entry with a uid of `2345` is sent to Partition 2. Any request for an entry with a uid outside the scope of the partitions defined will indicate that no such entry exists.

> **Note:**
>
> The upper boundary limit of a distribution algorithm is exclusive. This means that a search for uid `3000` in the example above returns an error indicating that the entry does not exist.

Consider the examples of searches using Numeric Distribution Algorithm.

The following search will be successful:

```
$ ldapsearch -b "uid=1010,ou=people,cn=example,cn=com" "cn=Ben"
```

However, the following searches will indicate that the entry does not exist (with result code `32`):

```
$ ldapsearch -b "uid=1010,o=admin,ou=people,cn=example,cn=com" "objectclass=*"

$ ldapsearch -b "uid=99,ou=people,cn=example,cn=com" "objectclass=*"
```

The following search will be broadcast, as the proxy cannot determine the partition to which the entry belongs, using the distribution algorithm defined above:

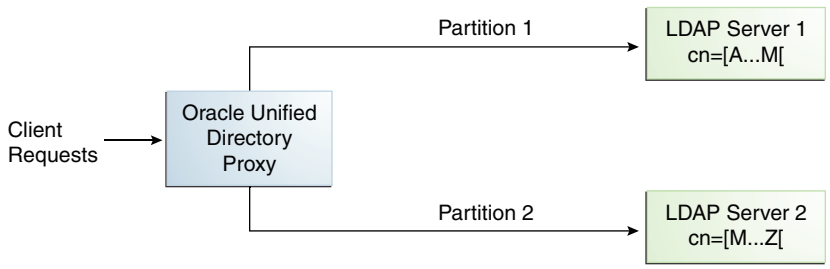
```
$ ldapsearch -b "ou=people,cn=example,cn=com" "uid=*"
```

12.3.3 Understanding Lexico Distribution

Understand Lexico distribution with the following example. With a distribution using lexico algorithm, the proxy forwards requests to one of the partitions, based on the alphabetical value of the first RDN after the distribution base DN in the request. When you set up distribution with lexico algorithm, you split the data of your database into different partitions, based on an alphabetical value of the attribute of your choice. The proxy then forwards all client requests to the appropriate partition, using the same algorithm.

For example, you could split your data into two partitions based on the cn of the entries, as illustrated in Figure 12-8.

Figure 12-8 Lexico Distribution Example



In this example, any requests for an entry with a cn starting with `B` such as `Ben` are sent to Partition 1, while requests for an entry with a cn from `M-Y` are sent to Partition 2.

> **Note:**
>
> The upper boundary limit of a distribution algorithm is exclusive. This means that a search for cn= `Zachary` in the example above will indicate that no such entry is found. If you want to include entries that start with Z in the search boundaries, then use the `unlimited` keyword. For example, use `cn=[M..unlimited[` to include all entries beyond M.

Consider the examples of searches using Lexico Distribution Algorithm:

The following search will be successful:

```
$ ldapsearch -b "cn=Ben,ou=people,cn=example,cn=com" "objectclass=*"
```

The following search will also be successful, as `cn=Ben` is the *first* RDN.

```
$ ldapsearch -b "uid=1010,cn=Ben,ou=people,cn=example,cn=com" "objectclass=*"
```

However, the following searches will indicate that the entry does not exist (with result code `32`):

```
$ ldapsearch -b "cn=Ben,o=admin,ou=people,cn=example,cn=com" "objectclass=*"

$ ldapsearch -b "cn=Zach,ou=people,cn=example,cn=com" "objectclass=*"
```

The distribution cannot determine to which partition the following search belongs and will be broadcast:

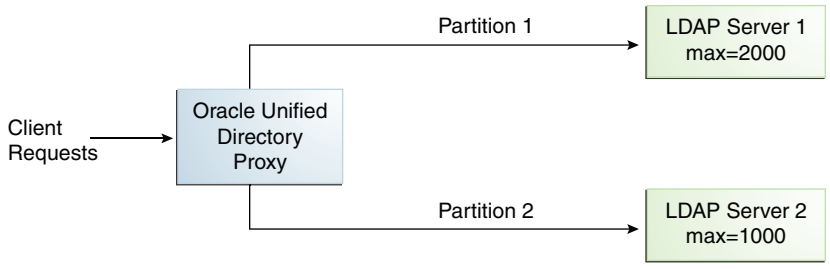
```
$ ldapsearch -b "ou=people,cn=example,cn=com" "cn=*"
```

12.3.4 Understanding Capacity Distribution

Understand Capacity distribution with the following example. With a capacity-based distribution, the proxy sends Add requests based on the capacity of each partition, which is determined by the maximum number of entries the partitions can hold. All other requests are distributed by the global index catalog or by broadcast.

Because the data is distributed to the partitions in a completely random manner, the easiest way to identify on which partition a particular data entry is by using a global index. Global index is mandatory when using capacity distribution. If no global index is set up, all requests other than Add will have to be broadcast. For more information about global indexes, see Understanding the Global Index Catalog and Configuring Global Indexes Using the Command Line.

Figure 12-9 Capacity Distribution Example



In the example illustrated in Figure 12-9, Partition 1 has twice the capacity of Partition 2, therefore Partition 1 will receive twice the add requests sent to Partition 2. This way, both partitions should be full at the same time. When all the partitions are full, the distribution will send one request to each partition at each cycle.

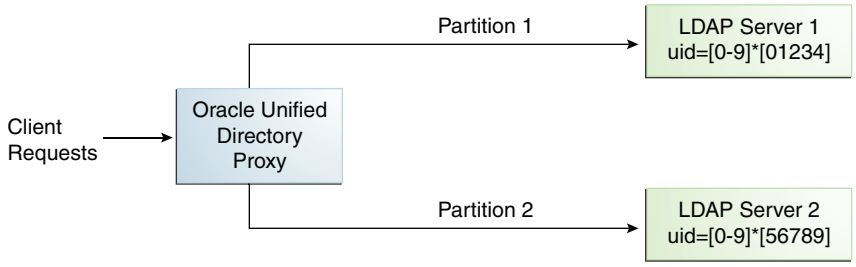
12.3.5 Understanding DN Pattern Distribution

Understand DN Pattern distribution with the following example. With a distribution using DN pattern algorithm, the proxy forwards requests to one of the partitions, based on the match between a request base DN and a string pattern. The match is only perform on the relative part of the request base DN, that is, the part after the distribution base DN.

For example, you could split your data into two partitions based on the DN pattern in the uid of the entries, as illustrated in Figure 12-10.

Distribution using DN pattern is more onerous than distribution with numeric or lexico algorithm. If possible, use another distribution algorithm.

Figure 12-10 DN Pattern Distribution Example

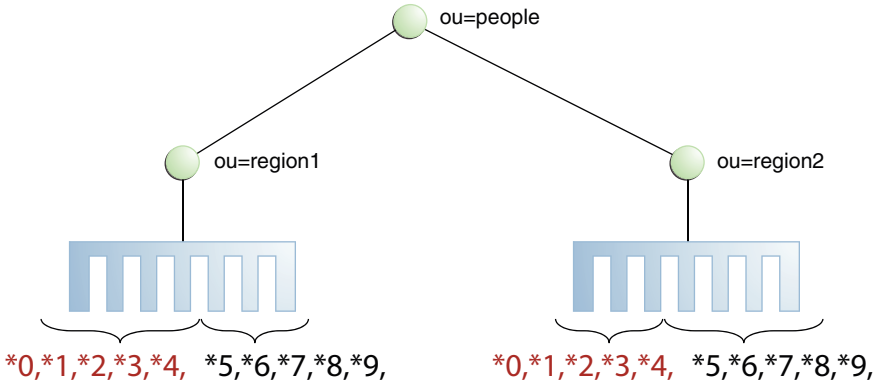


In this example, all the data entries with a uid that ends with 0, 1, 2, 3, or 4 will be sent to Partition 1. Data entries with a uid that ends with 5, 6, 7, 8, or 9 will be sent to Partition 2.

This type of distribution, although using numerical values is quite different from numeric distribution. In numerical distribution, the data is partitioned based on a numerical *range*, while DN pattern distribution is based on a *pattern* in the data string.

Distribution using a DN pattern algorithm is typically used in cases where the distribution partitions do not correspond exactly to the distribution base DN. For example, if the data is distributed as illustrated in Figure 12-11, the data for Partition 1 and Partition 2 is in both base DN `ou=people,ou=region1` and `ou=people,ou=region2`. The only way to distribute the data easily is to use the DN pattern.

Figure 12-11 Example of Directory Information Tree



Consider the example of DN Pattern Algorithm split by region.

If the deployment of the information is based in two geographical locations, it may be easier to use the DN pattern distribution to distribute the data. For example, if employee numbers were 4 digit codes, where the first digit indicated the region, then you could have the following:

Region 1	Region 2
1000	2000
1001	2001
1002	2002
1003	2003
1004	2004
1005	2005
1006	2006
1007	2007
1008	2008
1009	2009
1010	2010
...	...

To spread the data load, the entries in each location are split over two servers, where Server 1 contains all entries that end with 0, 1, 2, 3, and 4, while Server 2 contains all the entries that end with 5, 6, 7, 8, and 9, as illustrated in Figure 12-10.

Therefore, a search for DN pattern `1222` would be sent to partition 1, as would `2222`.

12.3.6 Understanding Union Workflow Element

The Union workflow element aggregate several DITs into a virtual unified DIT.

Topics:

- Overview of Union Workflow Element
- Configuration Parameters for Union Workflow Element

- Configuration Parameters for Union Partition

12.3.6.1 Overview of Union Workflow Element

The Union workflow element enables you to customize the bind and search routing policies.

For example, if an LDAP bind request satisfies multiple partitions to route the request to, then the Oracle Unified Directory always routes the request to the partition with highest priority. Consequently, authentication for users provisioned in other partition' data source would fail.

This workflow allows the bind operation to fall through the list of all eligible partitions that have been configured in this workflow. Similarly, it also provides a search policy that allows the search to continue to the next partition only when an entry is not returned from one partition.

12.3.6.2 Configuration Parameters for Union Workflow Element

Learn about the various Union workflow element configuration parameters, their names, descriptions, and functionality.

Property	Description	Mandatory or Optional	Value
`base-dn`	The base DN of the virtual unified DIT. A virtual unified DIT is an aggregation of one or more partitons.	Mandatory	A valid DN. For example, `ou=people,dc=example,dc=com`
`enabled`	This parameter Indicates whether the workflow element is enabled for use in the server. If a workflow element is not enabled, then its contents are not accessible when processing operations.	Mandatory	`true` or `false`
`cache-size`	Defines the maximum number of entries that can be stored in the cache used to filter out entry duplicates. When the entry duplicates filtering is active, the Union workflow element stores the returned entry DNs in a cache. A cache is specified to a search operation, and when the cache is full, the search operation is aborted. By default, the cache can handle `10000` DNs. When the cache size is set to `0` or is negative, then no limit is enforced.	Optional	`10000`

Property	Description	Mandatory or Optional	Value
`bind-option`	This parameter that supports the following bind options: • `bind-first-success`: Allows the bind to fall through the list of all eligible partitions (configured in this workflow element) until it reaches a first success. • `bind-quick-fail`: Causes the bind to immediately throw an error if multiple users from different partitions (configured in this workflow element) have the same DN that is used as a bind DN. Specify this option if you want the bind to fail for multiple users having the same DN from different configured partitions.	Optional	• `bind-first-success` (default) • `bind-quick-fail`
`search-first-match`	Indicates if a subtree search should stop the processing once an entry is found. A subtree search with the filter (`uid=user.1`) is likely to target a single entry. However, in some deployment, there can be several entries with `uid=user.1` in the various data sources. To prevent the union workflow element from returning more than one entry with `uid=user.1`, set this property to `true`.	Optional	• `true` (default value) • `false`
`auto-tune-search-option`	A parameter that controls whether the Union workflow element has the intelligence to switch to SearchAllCandidates mode. (This parameter is disabled or set to false, by default) In certain cases, this workflow element has the intelligence to switch to SearchAllCandidates mode, even if it is configured to SearchFirstMatch when the search filter mandates to search in all candidates. For example, the workflow element can switch modes if the search filter contains non-equality components (such as cn=*), if all of the components are objectclass attributes (such as objectclass=*), and so forth.	Mandatory	• `false` (default mode): It disables the workflow element's ability to switch modes. • `true` : If specified, then enables the workflow element to switch to SearchAllCandidates mode.

12.3.6.3 Configuration Parameters for Union Partition

The Union Partition is a subtree in the virtual unified DIT of a union workflow element. Learn about the various Union workflow element configuration parameters, their names, descriptions, and functionality.

Property	Description	Mandatory or Optional	Value
`allowed-operations`	Defines The list of operations that can be executed in this partition. By default, all the operations are allowed in the partitions.	Optional	• `add`: Allows add operation in this partition. • `bind`: Allows bind operation in this partition. • `compare`: Allows compare operation in this partition. • `delete`: Allows delete operation in this partition. • `extended`: Allows extended operation in this partition. • `modify`: Allows modify operation in this partition • `modifydn`: Allows modifyDN operation in this partition. • `search`: Allows search operation in this partition.
`black-list-attributes`	The list of attributes of DN syntax that must NOT be transformed. The `black-list-attributes` specifies the set of attributes that must not be transformed. This attribute is mutually exclusive with `white-list-attributes`, that is only one of them may be used in the configuration entry.	Optional	List off attributes with a DN Syntax

Property	Description	Mandatory or Optional	Value
`priority`	The priority of the partition. The priority of the partition is used to determine the order in which a request is sent to several partitions. The priority is a positive integer - the lower the value, the higher the priority, and 0 is the highest priority a partition can ever have.	Optional	An integer value. Lower value is.`0` Default value is `10`.
`relative-base-dn`	The relative part of the partition base DN. The relative base DN is a sequence of RDNs, and is used to build the partition base DN (The partition base DN is the concatenation of the relative base DN and the union base DN). For example if the relative base DN is set to `ou=people`, and the union base DN is `dc=example,dc=com`then the partition base DN will be `ou=People,dc=example,dc=com`. **Note:** The partition base DN is the naming context of the partition in the virtual DIT. The default value for the relative base DN is the null suffix (""). This means that the partition base DN is equal to the union base DN.	Optional	A valid DN.
`source-base-dn`	The base DN for the source DIT. The source DIT is the place from which entries are read (For example, remote LDAP server) to populate the union partition.. The source base DN need not to be equal to the partition base DN. When the source base DN is different than the partition base DN, a DN mapping is performed silently, and all the attributes with a DN syntax are automatically mapped accordingly. If you want to fine tune which attributes should, or should not be mapped, then you must set the `white-list-attributes` or the `black-list-attributes`.	Optional	A valid DN.

Property	Description	Mandatory or Optional	Value
`white-list-attributes`	The list of attributes with DN syntax that must be transformed. The `white-list-attributes` specifies the set of attributes that must be transformed. This attribute is mutually exclusive with `black-list-attributes`, that is only one of them may be used in the configuration entry. **Note:** If the `black-list-attribute` property is not specified then, all attributes with a DN syntax are transformed	Optional	List off attributes with a DN Syntax
`workflow-element`	The partition workflow element. The workflow element to use as a data source to populate the partition.	Mandatory	The DN of any Workflow Element. Ensure that the `enabled` configuration property of the referenced workflow element is set to `true`.

12.4 Understanding Data Integration Using the Proxy

You can use different methods to retrieve and integrate data from a variety of sources, including databases and directories, to present a unified view of that data.

The following topics describes the different methods:

- Understanding How to Retrieve All Attribute Values from an Active Directory Server
- About Enterprise User Security Databases Integration
- Overview of Enabling LDAP Clients to Update User Passwords Stored in Active Directory
- Understanding Pass-Through Authentication
- Understanding Oracle Unified Directory Plug-Ins
- Overview of Transforming Remote LDAP Server's Global Unique Identifier Value

> **Note:**
>
> For information about configuring data integration, see Configuring Integration Using the Proxy .

12.4.1 Understanding How to Retrieve All Attribute Values from an Active Directory Server

You can retrieve the complete set of values for the attribute from an active directory server by performing multiple search requests, each retrieving a distinct subset.

Retrieving the contents of a multi-valued attribute sometimes result in a large number of returned values. Microsoft Active Directory server limits the maximum number of attribute values that can be retrieved in a single query.

Microsoft Active Directory provides a **range retrieval** mechanism that allows you to retrieve all the values of a multi-valued attribute. This mechanism permits a client-specified subset of the values to be retrieved in a search request. By performing multiple search requests, each retrieving a distinct subset, the complete set of values for the attribute can be retrieved.

Oracle Unified Directory handles Active Directory range retrieval by providing support for Microsoft Active Directory paging. The main purpose of Microsoft Active Directory paging is to detect if a range option is present among the options of the returned attributes and to retrieve the complete range of attribute values from the Microsoft Active Directory server. This complete set of attribute values is returned, so that the client application does not have to deal with the range option.

Microsoft Active Directory paging is implemented as a workflow element that is relevant only if the leaf of the workflow element chain is connected to an Active Directory server. You can configure the following properties of an Active Directory Paging workflow element:

- The next workflow element in the chain as this workflow element is not a leaf workflow element
- An optional list of attributes that can reduce the processing of scanning attributes to detect the range option

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

> **Note:**
>
> For information about how to configure Microsoft Active Directory paging, see Retrieving All Attribute Values from an Active Directory Server.

12.4.2 About Enterprise User Security Databases Integration

You can integrate Oracle Unified Directory and Enterprise User Security to leverage user identities stored in an LDAP-compliant directory service without any additional synchronization.

When integrated with Enterprise User Security, Oracle Unified Directory supports the following:

- Microsoft Active Directory
- Novell eDirectory
- Oracle Unified Directory
- Oracle Directory Server Enterprise Edition

For more information about Oracle Enterprise User Security, see the *Oracle Database Enterprise User Security Administrator's Guide*. For detailed instructions on configuring Oracle Unified Directory and Enterprise User Security to work together, see Integrating Oracle Unified Directory with Oracle Enterprise User Security.

12.4.3 Overview of Enabling LDAP Clients to Update User Passwords Stored in Active Directory

The Ad Password workflow element enables Oracle Unified Directory LDAP client applications to update user passwords stored in Microsoft Active Directory and Active Directory Lightweight Directory Services (AD LDS) using the LDAP protocol.

This section includes the following topics:

- About Ad Password Workflow Element
- Understanding Ad Password Workflow Element Functionality
- About Ad Password Workflow Element Check for an SSL Connection
- Considerations for Using the Ad Password Workflow Element

To configure an Ad Password workflow element, see Updating User Passwords Stored in Active Directory.

12.4.3.1 About Ad Password Workflow Element

Microsoft Active Directory and AD LDS have characteristics and requirements that Oracle Unified Directory LDAP clients cannot always handle using standard LDAP operations.

For example, if a client updates a user password (`userPassword` attribute) using a standard LDAP modify operation, the update is successful on most LDAP servers. Active Directory will accept this modify operation, but it will not update the user password because of the following requirements:

- Active Directory stores a user password in the `unicodePwd` attribute on a user object rather than in the `userPassword` attribute.

  The syntax for the `unicodePwd` attribute is an octet-string containing a UNICODE string enclosed in double quotes (").
- The `unicodePwd` attribute cannot be added during the creation of a user object. The user object must first be created without the `unicodePwd` attribute and then the attribute is added on the new object with a modify operation.

- Only an administrator can reset a user password without knowing the previous password.
- Active Directory user passwords can be updated only over an SSL connection.

The Ad Password workflow element can handle these specific requirements. It allows existing client applications to update user passwords stored in Active Directory or AD LDS using standard LDAP operations, without requiring the client applications to be re-coded.

For the supported versions of Active Directory and AD LDS, check the Certification Matrix.

12.4.3.2 Understanding Ad Password Workflow Element Functionality

The Ad Password workflow element performs specific functions, depending on the LDAP operation it is processing.

This section includes the following topics:

- Understanding Ad Password Workflow Element Mechanism for ADD Operations
- Understanding Ad Password Workflow Element Mechanism for MODIFY Operations

12.4.3.2.1 Understanding Ad Password Workflow Element Mechanism for ADD Operations

If a secure proxy LDAP workflow element is configured, the Ad Password workflow element handles an ADD operation that contains a `userPassword` attribute as follows:

- Maps the `userPassword` attribute to the `unicodePwd` attribute (`map-userpassword` property is set to `true`).
- Handles the ADD operation that contains a `userPassword` attribute in the following order:
  1. Executes an ADD operation on the Active Directory user object without the `unicodePwd`, `useraccountcontrol`, and `msds-useraccountdisabled` attributes. This operation is handled by the workflow element defined by the `next-workflow-element` property in the Ad Password workflow element.
  2. Executes a MODIFY operation on the user object to create the `unicodePwd` attribute. This operation is handled by workflow element defined by the `secure-proxy-workflow-element` property in the Ad Password workflow element.
  3. If the original ADD operation contained a `useraccountcontrol` or `msds-useraccountdisabled` attribute, executes a MODIFY operation on the user object. This operation is handled by the workflow element defined by the `next-workflow-element` property in the Ad Password workflow element.

     The `useraccountcontrol` and `msds-useraccountdisabled` attributes cannot be set before the `unicodePwd` attribute is created in Step 2.

If Step 2 or Step 3 fails either during the bind or the MOD operation, the ADD operation is rolled back (that is, the entry is deleted).

If a secure proxy LDAP workflow element is **not** configured, the Ad Password workflow element handles an ADD operation that contains a `userPassword` attribute as follows:

- Maps `userPassword` to `unicodePwd`, if needed (`map-userpassword` property is set to `true`).
- Handles the ADD operation by the workflow element defined by the `next-workflow-element` property in the Ad Password workflow element. If the `next-workflow-element` does not use SSL, then Active Directory might refuse the operation.

12.4.3.2.2 Understanding Ad Password Workflow Element Mechanism for MODIFY Operations

If a secure proxy LDAP workflow element is configured, the Ad Password workflow element handles a MODIFY operation that contains a `userPassword` attribute as follows:

1. Executes a MODIFY operation on the user object with the user password change. This operation is handled by the workflow element defined by the `secure-proxy-workflow-element` property in the Ad Password workflow element.

   If this step fails either during the bind or the MOD operation, Oracle Unified Directory returns the MOD result code to the client without trying Step 2.

2. Executes the MODIFY operation on the object. This operation is handled by the workflow element defined by the `next-workflow-element` property in the Ad Password workflow element.

If a secure proxy LDAP workflow element is **not** configured, the Ad Password workflow element handles a MODIFY operation that contains a user password as follows:

- Maps `userPassword` to `unicodePwd`, if needed (`map-userpassword` property is set to `true`).
- Executes a MODIFY operation on the object. This operation is handled by the workflow element defined by the `next-workflow-element` property in the Ad Password workflow element.

12.4.3.3 About Ad Password Workflow Element Check for an SSL Connection

When SSL is required, the Ad Password workflow element checks that an SSL connection is configured to the remote Active Directory or AD LDS server, as follows:

- If you configure a `secure-proxy-workflow-element`, Oracle Unified Directory will check that this workflow element is a proxy LDAP workflow element that is using an LDAP server extension configured to always use SSL (`remote-ldap-server-ssl-policy` property set to `always`).
- If you do not configure a `secure-proxy-workflow-element`, the `next-workflow-element` must use an LDAP server extension configured to always use SSL.

If operations on a user password fail because of an incorrect configuration, the Ad Password workflow element returns the error codes it receives from the remote Active Directory or AD LDS server.

12.4.3.4 Considerations for Using the Ad Password Workflow Element

Before you create and configure an Ad Password workflow element, consider your deployment's security and performance requirements using the following use cases:

- About All LDAP Operations Over an SSL Connection Configuration
- About Only LDAP Operations for Password Modifications Over an SSL Connection Configuration

> **See Also:**
>
> - Understanding Ad Password Workflow Element Functionality for information about how LDAP ADD and MODIFY operations are handled.
> - Configuring Security Between the Proxy and the Data Source for more information about the security between a proxy and a data source such as Active Directory or AD LDS server.

12.4.3.4.1 About All LDAP Operations Over an SSL Connection Configuration

This use case performs all LDAP operations between clients and Active Directory or AD LDS server over an SSL connection.

The advantage of this use case is that all LDAP operations are always performed over a fully secure SSL connection, regardless of how the client connects to the proxy server. A disadvantage is that some LDAP operations performed over an SSL connection can cause performance degradation for your deployment.

Configuration Requirements

This use case requires the following components:

- An LDAP server extension configured with the `remote-ldap-server-ssl-policy` property set to `always`.
- A secure proxy LDAP workflow element that points to an LDAP server extension as described in the previous item (that is, configured with the `remote-ldap-server-ssl-policy` option set to `always`).
- An Ad Password workflow element configured with the `next-workflow-element` property pointing to a secure proxy LDAP workflow element.

12.4.3.4.2 About Only LDAP Operations for Password Modifications Over an SSL Connection Configuration

This use case performs operations that are related to password modifications over an SSL connection to Active Directory or AD LDS server. Other LDAP operations are performed over an SSL or non-SSL connection according to the `remote-ldap-server-ssl-policy` configuration property of the LDAP server extension used by `next-workflow-element`.

An advantage of this use case is that it forces password modifications to take place over an SSL connection, without requiring all communications to the remote server to use SSL. The other communications can either never use SSL or use SSL only if the client connection is using SSL.

Configuration Requirements

This use case requires the following components:

- Two LDAP server extensions to communicate with the remote Active Directory or AD LDS server:
  - An LDAP server extension for SSL connections. The `remote-ldap-server-ssl-policy` property must be set to `always`.

  - Another LDAP server extension for operations not related to password modifications. The `remote-ldap-server-ssl-policy` property is set to either `never` or `user` (or omitted).
- Two proxy LDAP workflow elements to communicate with the remote Active Directory or AD LDS server:
  - A secure proxy LDAP workflow element for SSL connections.
  - Another proxy LDAP workflow element for operations not related to passwords.
- Ad Password workflow element configured with both the `secure-proxy-workflow-element` and `next-workflow-element` properties:
  - LDAP operations that modify the user password are handled by the workflow element specified by the `secure-proxy-workflow-element` property and will take place over an SSL connection.
  - Other LDAP operations not related to password modifications are handled by the workflow element specified by the `next-workflow-element`.

12.4.3.4.3 About Active Directory Configuration When Attribute Mapping is Not Required

With a specific configuration, Active Directory and AD LDS can handle modifications on the `userPassword` attribute without requiring the `userPassword` attribute to be mapped to the `unicodePwd` attribute.

This Active Directory or AD LDS configuration requires:

- The domain controller (DC) must be running as Active Directory or AD LDS, and the domain functional level must be Windows 2003 or greater.
- The `fUserPwdSupport` character must be set to `true` in the `dSHeuristics` attribute.

For more information about this specific configuration, see the following Microsoft document:

`http://msdn.microsoft.com/en-us/library/cc223249.aspx`

For this configuration, user password attribute mapping is not required. To control this mapping, the Ad Password workflow element provides the `map-userpassword` property:

- `true` (default) enables mapping. The `userPassword` attribute is automatically mapped to `unicodePwd`. LDAP ADD and MODIFY operations are then performed on `unicodePwd` instead of `userPassword`.
- `false` disables mapping. LDAP ADD and MODIFY operations are performed on `userPassword`.

12.4.4 Understanding Pass-Through Authentication

Pass-through authentication (PTA) is a mechanism where one directory server consults another directory server to authenticate bind requests. A typical scenario for pass-through authentication involves passing authentication through to Active Directory for users coming from Oracle Unified Directory.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

The following topics describe the use and operation of pass-through authentication:

- Overview of the Pass-Through Authentication Mechanism
- Understanding the Pass-Through Authentication Configuration Model
- Understanding the Pass-Through Authentication Configuration Parameters
- Overview of Pass-Through Authentication Implementation for Different Servers
- Understanding Implementation of Pass-Through Authentication for a Kerberos Server

> **Note:**
>
> For information about configuring pass-through authentication, see Overview of Configuring Pass-Through Authentication.

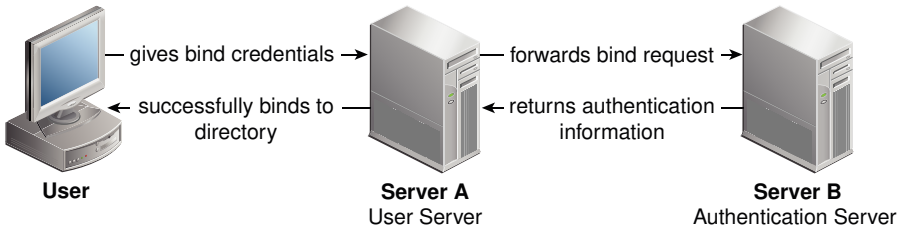
12.4.4.1 Overview of the Pass-Through Authentication Mechanism

You use the pass-through authentication mechanism when the client attempts to bind to the directory server and the user credentials for authenticating are not stored locally, but instead in another remote directory server known as the authentication (Auth) server. The directory server then redirects the bind operation to the authentication server to verify the credentials. The credential here refers to the `userpassword` attribute. The Auth server that stores the user credentials can be Oracle Unified Directory, Microsoft Active Directory, or an LDAP V3 compliant directory server.

Exactly how Oracle Unified Directory redirects the bind depends on how the user entry in user server maps to the corresponding user entry in the authentication server. Oracle Unified Directory supports one-to-one mapping between the user entry and the authentication entry.

To gain a better understanding of the pass-through authentication mechanism, consider the example depicted in Figure 12-12.

Figure 12-12 Pass-Through Authentication Mechanism



Let us consider two servers, say server A and server B and a user entry `cn=myuser` stored on server B. Now, if a user attempts to access server A to perform any operation it has to first bind

to server A with its credentials for authentication. However, the credentials are not present on server A, therefore the bind to server A would normally fail. But, using the pass-through authentication mechanism, server A can verify the credential by directing the bind request to server B. After the credentials are validated using server B, and the bind is successful then server A returns success for the bind operation.

The Server A in this example acts as the user directory server or the pass-through authentication directory server. This is because it is the server that passes the bind request to another directory server. The authentication directory server B, acts as the authenticating directory, the server that contains the entry and verifies the bind credentials of the requesting client.

12.4.4.2 Understanding the Pass-Through Authentication Configuration Model

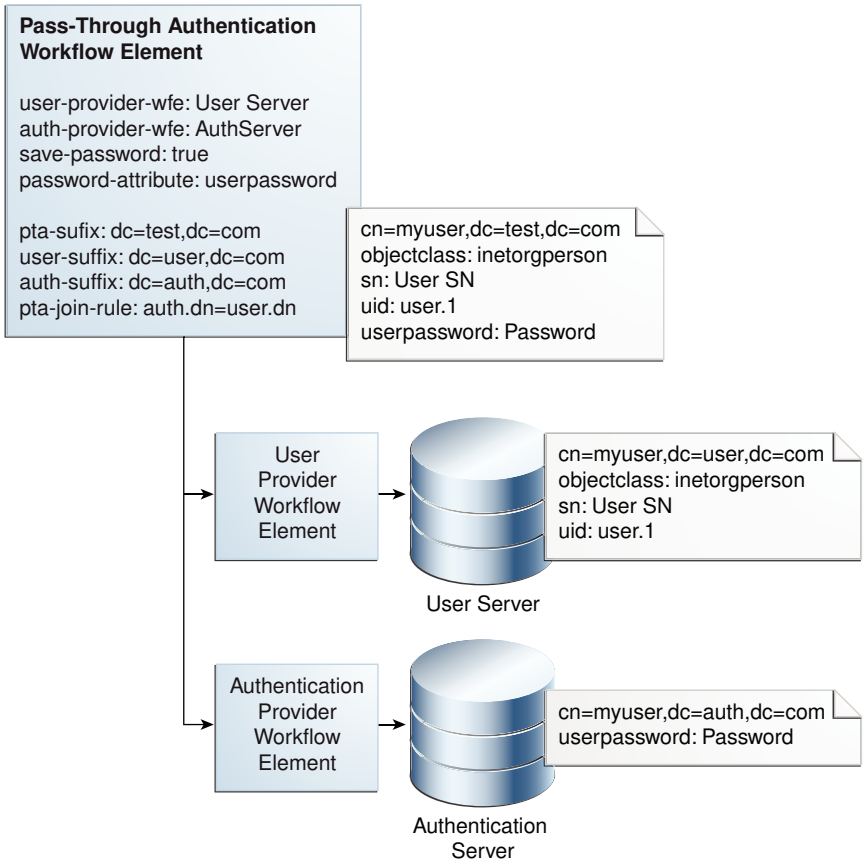
Oracle Unified Directory implements pass-through authentication using pass-through authentication workflow element that allows you to administer your user and authentication directories on separate instances of directory server.

The user provider is a workflow element that contains the user entries, which is to say all attributes except the password of the user. On the other hand, the authentication provider is the workflow element that contains the user password.

> **Note:**
>
> Oracle Unified Directory provides support for local back end or proxy for both user provider workflow element and authentication provider workflow element. However, Kerberos is supported for authentication provider workflow element only.

Figure 12-13 illustrates a pass-through authentication configuration model.

Figure 12-13 Pass-Through Authentication Configuration Model



12.4.4.3 Understanding the Pass-Through Authentication Configuration Parameters

Table 12-1 describes the configuration parameters used in the pass-through authentication configuration model described in Understanding the Pass-Through Authentication Configuration Model.

For more information about configuring pass-through authentication using `dsconfig` command, see Overview of Configuring Pass-Through Authentication.

For more information about configuring pass-through authentication using OUDSM, see Creating a Workflow Element.

Table 12-1 Configuration Parameters Used in Pass-Through Authentication Process

Parameter	Description
`user-provider-workflow-element`	User Provider Workflow Element This parameter defines the workflow element that contains the user entries. This workflow element is used for all the operations except BIND. This is a mandatory parameter.

Table 12-1 (Cont.) Configuration Parameters Used in Pass-Through Authentication Process

Parameter	Description
`auth-provider-workflow-element`	Authentication Provider Workflow Element This parameter defines the workflow element that contains the authentication entry and verifies the bind credentials of the requesting client. This workflow element is used for `BIND` and `MODIFY` operations on the `userpassword` attribute. This is a mandatory parameter.
`save-password-on-successful-bind`	This parameter allows you to enable or disable the password-copy feature. If this parameter is set to `true`, and the BIND on the authentication provider workflow element succeeds then a copy of the password is saved in the user provider workflow element. The copy is a MODIFY operation applied on the DN used for the bind, replacing the value `userpassword` with the value of the password used for the bind. This is an optional parameter. The default value is `false`.
`password-attribute`	This parameter defines the attribute in which the password value is copied in the user entry when the password-copy feature is enabled. After the password is saved, it can be copied in the `userpassword` attribute or in another attribute in the user provider workflow element. This is an optional parameter. The default value is `userpassword`.
`pta-suffix`	This parameter defines the virtual DN that is exposed by the pass-through authentication workflow element. This is an optional parameter. By default, this parameter is not set and implies that there is no DN mapping.
`user-suffix`	This parameter defines the actual suffix that contains the user entries on the user provider workflow element. This is an optional parameter. By default, this parameter is not set and implies that the DN is the same as the `pta-suffix` parameter.
`auth-suffix`	This parameter defines the actual suffix that contains the authentication entries on the authentication provider workflow element. This is an optional parameter. By default, this parameter is not set and implies that the DN is the same as the `pta-suffix` parameter.
`pta-join-rule`	This parameter defines the mapping between an authentication entry and a user entry. This is an optional parameter. By default, this parameter is not set and implies that the rule is `auth.dn=user.dn`.

12.4.4.4 Overview of Pass-Through Authentication Implementation for Different Servers

If your user entries are stored on a Kerberos server, then you must configure a Kerberos workflow element.

This section contains the following topics:

- Understanding Pass-Through Authentication Workflow Element Features
- Considerations for Using Pass-Through Workflow Element
- Handling LDAP Operations Using Pass-Through Authentication Workflow Element

See Configuring Pass-Through Authentication for Different Servers for more information.

12.4.4.4.1 Understanding Pass-Through Authentication Workflow Element Features

The following is a list of some pass-through authentication workflow element capabilities:

- Allows you to route requests to a specific workflow element depending on the request type. For instance, bind requests are routed to authentication workflow element. When you apply `MODIFY` on any attribute except `userpassword` it is routed to the user workflow element. Applying `MODIFY` on the `userpassword` attribute is routed to the authentication workflow element (and also to the user workflow element if password-copy is enabled). All other requests, such as `ADD, DELETE, RENAME, COMPARE,` and `SEARCH` are routed to the user workflow element.
- Support for Kerberos workflow element as an authentication workflow element. When the authentication workflow element is a Kerberos workflow element, Oracle Unified Directory forwards the authentication request to a Kerberos server, and the authentication is performed using Kerberos protocol instead of LDAP bind.
- Simplifies migration from an external LDAP server containing user credentials to Oracle Unified Directory. During the migration phase, the pass-through authentication workflow element copies the user password from the external LDAP server to Oracle Unified Directory on successful bind. This feature is called as password-copy. For instance, when a user successfully authenticates, the bind is routed to the authentication workflow element, which is the external LDAP server. The pass-through authentication workflow element then stores the password used for this bind operation in the user workflow element. This migration phase populates the user password attribute of all the users that initiated contact during the migration phase.
- Support cases where the entry on the authentication workflow element is linked to the entry on the user workflow element by a join rule and an authentication suffix. This join rule can be a `DN=DN` mapping or a simple join rule with the following format:

  ```
  auth.<Attribute1>=user.<Attribute2>
  ```

  For more information about join rules, see Overview of Join Rules.

  The mapping between the user entry and the authentication entry must be a one-to-one mapping, which implies that each entry in the user provider corresponds with one entry in the authentication provider.
- Support for DN mapping, for instance allows you to publish entries below `dc=pta,dc=com` whereas the user workflow element suffix is `dc=user,dc=com`.
- Support for password modification.
- Support for all kinds of workflow element for the user workflow element, local or remote.

12.4.4.4.2 Considerations for Using Pass-Through Workflow Element

When using the pass-through authentication workflow element, you must keep the following in mind:

- The authentication workflow element handles only bind requests.
- The user provider workflow element is used for all other operations, such as `ADD, DELETE, RENAME, COMPARE,` and `SEARCH`.
- The `MODIFY` operation depends on the `save-password-on-successful-bind` parameter. This parameter saves the password if needed in the user workflow element when pass-

through authentication workflow element binds successfully with the authentication workflow element.

If `save-password-on-successful-bind` is enabled, then `userpassword` parameter is modified on both participants.

If `save-password-on-successful-bind` is disabled, then the userpassword is modified on the authentication participant only.

- If you define the `user-suffix` or `auth-suffix` parameter, then you must define the `pta-suffix`. Both parameters apply to DN renaming between the user or authentication participant and the pass-through authentication participant.
- If a join rule is defined, and the authentication and user entries do not necessarily have the same DN, then you must define the `auth-suffix`.
- Note, if `user-suffix` is not defined, then the workflow element assumes that the `user-suffix=pta-suffix`. The same applies if the `auth-suffix` not defined. Here, again the workflow element assumes that the `auth-suffix=pta-suffix`.

12.4.4.4.3 Handling LDAP Operations Using Pass-Through Authentication Workflow Element

Oracle Unified Directory supports the following LDAP operations using pass-through authentication workflow element:

Operation	Description
ADD	All `ADD` operations processed through the pass-through authentication workflow element are sent to the user provider workflow element. • If the save-password-on-successful-bind parameter is set to `true`, then the `userpassword` attribute is also stored in the user-provider workflow element. • If the feature is disabled, then the `userpassword` attribute is not stored in the user-provider workflow element.
BIND	The `BIND` operation is routed to the authentication-provider workflow element. • If the `BIND` is successful and the `save-password-on-successful-bind` parameter is enabled, the pass-through authentication workflow element also tries to attempt a `BIND` on the user-provider workflow element to check if there is a local copy of the password. • If the `BIND` fails, then the `userpassword` attribute is copied to the user-provider workflow element.
COMPARE	The `COMPARE` operation is routed to the user-provider workflow element. The `COMPARE` operation that is applied to the `userpassword` attribute is routed to the user-provider workflow element, which may not contain the attribute unless the `save-password-on-successful-bind` parameter is enabled.
DELETE	The `DELETE` operation is routed to the user-provider workflow element only. The entry on the authentication server is not deleted.
MODIFY	For all attributes except `userpassword`, the modifications are performed on the user-provider workflow element. For the `userpassword` attribute: • If `save-password-on-successful-bind` parameter is enabled, then the password is modified on both the user-provider workflow element and the authentication-provider workflow element. • If `save-password-on-successful-bind` parameter is disabled, the password is modified on the authentication-provider workflow element only. • If the authentication provider is a Kerberos workflow element, then the modify password operations fails.

Operation	Description
MODIFY_DN	The pass-through authentication workflow element processes `MODIFY_DN` on the user-provider workflow element only and does not modify the entry on the authentication-provider workflow element.
SEARCH	The `SEARCH` operations are routed to the user-provider workflow element only. This in turn implies, that a `SEARCH` operation that submits a request for the `userpassword` attribute might not return any value unless there is a copy in the user-provider workflow element.

12.4.4.5 Understanding Implementation of Pass-Through Authentication for a Kerberos Server

If your user entries are stored on a Kerberos server, then you must configure a Kerberos workflow element. See Configuring Pass-Through Authentication for a Kerberos Server for more information.

12.4.5 Understanding Oracle Unified Directory Plug-Ins

Oracle Unified Directory provides a plug-in API that enables you to extend existing directory server functionality. A plug-in is similar to a workflow element and you can insert a plug-in into any Oracle Unified Directory workflow element tree.

You may want to develop your own plug-ins when you have a particular directory server requirement and Oracle Unified Directory does not provide the necessary functionality to accommodate that requirement.

For more information about Oracle Unified Directory plug-ins, see "Understanding Basic Oracle Unified Directory Plug-in Concepts" in the Oracle Fusion Middleware Developing Plug-Ins for Oracle Unified Directory.

12.4.6 Overview of Transforming Remote LDAP Server's Global Unique Identifier Value

All LDAP repositories contain a global unique identifier (GUID) for every entry. Many Oracle applications rely on the `orclguid` attribute as the unique identifier for entries returned by OUD proxy.

LDAP directories, such as Active Directory and eDirectory store their LDAP entry's GUID in a binary format. When configuring these remote backends, OUD proxy can perform transformation between the corresponding binary value from the remote LDAP server's GUID attribute and the `orclguid` attribute for both inbound and outbound search operations. This however, is not configured by default. To achieve this transformation, you need to configure the `remote-ldap-server-guid` property in LDAP server extension. See Modifying the Advanced Properties of an LDAP Server Extension.

12.5 Understanding Virtualization

You can view and retrieve data from virtual directories and data sources using different features.

The following sections describe how different Oracle Unified Directory features enable you to view and retrieve data from virtual directories and data sources:

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

- Using Entries from Multiple Directories
- Overview of Optimizing Search Results From Virtual Directories Using Workflow Elements
- Understanding Addition of `memberof` User Attributes to `person` Entries
- Overview of Renaming DNs Using the Proxy
- Understanding How to Modify RDN Values Using the Proxy
- Understanding How to Retrieve Attributes from a SAML Identity Provider Using SAML XASP
- Understanding ForkJoin Workflow Element

12.5.1 Using Entries from Multiple Directories

You can understand about the Join workflow element, which presents a virtual directory view of your repositories and routes data to and from those repositories from the following sections.

This section covers the following topics:

- Understanding the Join Workflow Element
- Understanding Join Participants
- Overview of Join Rules
- Overview of Join Policies
- Understanding Supported Joiner Types
- Understanding the Join Condition
- About Virtual Attributes Creation
- Overview of Attribute Flow Settings
- About Bind Operations
- About DN Attributes Translation
- Configuring the Criticality of Join Participants
- Understanding Enabled Operations
- Understanding How to Cascade Write Operations to Secondary Participants

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

12.5.1.1 Understanding the Join Workflow Element

For most enterprises, user identity information such as user profiles, access data, and authorization data for a single entry is scattered across heterogeneous data sources at multiple locations. For example, employee information is stored in HR databases or in Microsoft Active Directories, customer and partner data in CRM databases, and additional LDAP directories. Companies require aggregated user data from various data sources in real time. As a consequence, application-specific directories proliferate, copying and synchronizing identity data, which leads to high administration and maintenance costs, inconsistent identity data, and compliance issues.

Oracle Unified Directory provides a directory service solution that addresses these challenges. Oracle Unified Directory supports the Join workflow element that presents a virtual directory view of the repositories and then routes data to and from the repositories.

Oracle Unified Directory enables you to define workflow elements, such as the Proxy LDAP workflow element, to connect to its underlying data repositories. The Join workflow element enables you to combine data from different workflow elements, as needed, to present a customized directory tree.

The Join workflow element is dynamic and does not require synchronization between its data sources. It consolidates identity data without moving data from its native locations, and reuses identity data without copying. These capabilities lead to ease of deployment, diminished costs, simplified identity infrastructure, and a high return on investment by eliminating the need to constantly adapt the applications from changes in the identity stores.

> **Note:**
>
> Be aware that directory virtualization is not running a directory server in a virtualized environment.

When data corresponding to a single entry is spread across multiple data sources, this workflow element combines those different data sources into one unified LDAP view, which is similar to a relational database's table join. Join workflow element does not connect to the underlying data repository. Instead, it builds on top of one or more proxy sources or local back ends to assemble its data as needed. Think of the Join workflow element as joining two or more data repositories by defining Join relationships, known as *Joiners*, between workflow elements. You can configure as many workflow elements as needed.

> **Note:**
>
> Do not confuse Join with Distribution.
>
> - Use Distribution when some entries are on server A and others on server B. A given entry is stored inside a single server, either A or B, with all its attributes.
> - Use Join when a part of the entry (such as some attributes) is stored on server A and another part is stored on server B.
>
> For more information about the Distribution workflow element, see Configuring a Distribution Deployment Using the `dsconfig` Command.

12.5.1.1.1 About Join Workflow Element Features

Following are key features of the Join workflow element:

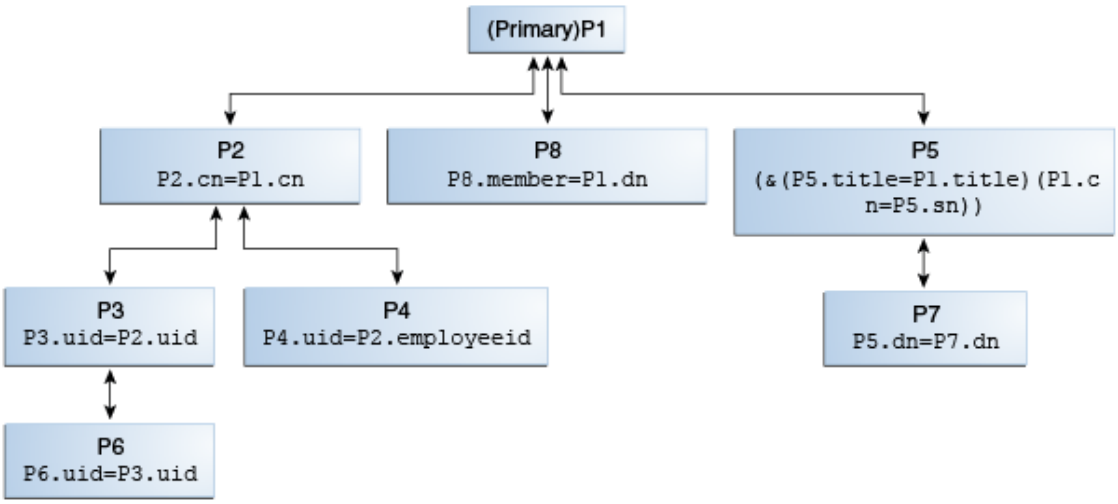
- Allows you to define a relationship between any two participating elements. Supports one primary participant and any number of secondary participants. For more information, see Understanding Join Participants.
- Supports a sophisticated relationship tree among Join participants using complex Join rules. For more information, see Overview of Join Rules.
- Queries all associated secondary participants for each entry retrieved from the primary participant to form the combined entry. For more information, see Overview of Join Rules.
- Adds a `joinedentrydn` attribute value to each entry retrieved from one participating element, indicating which entries from secondary participants were used to form the consolidated entry. For more information, see Overview of Join Rules.
- Supports different Joiner types, such as one-to-one, many-to-one, and shadow for different kinds of Join scenarios. For more information, see Understanding Supported Joiner Types.
- Allows you to merge attributes and objectclasses from multiple participants to form a new virtual entry. For more information, see About Virtual Attributes Creation.
- Allows you to specify which attributes can be retrieved and which the attributes can be stored in a participating data source. For more information, see Overview of Attribute Flow Settings.
- Supports bind-fall through feature. For more information, see About Bind Operations.
- Supports translation of DN-syntaxed attribute values from each repository suffix to a common Join workflow element suffix. For information, see About DN Attributes Translation.
- Allows you to configure criticality of Join participants. For more information, see Configuring the Criticality of Join Participants.
- Supports operations that are set as enabled. For more information, see Understanding Enabled Operations .
- Allows you to cascade write operations. For more information, see Understanding How to Cascade Write Operations to Secondary Participants.

12.5.1.1.2 Understanding the Join Workflow Element Configuration Model

Figure 12-14 illustrates the configuration model for the Join workflow element and Join *participants* using *Join rules*.

A participant is a workflow element that contributes information to the Join workflow element to form a combined joined entry. Join rules determine how an entry from one participant relates to an entry from another participant.

Figure 12-14 Join Workflow Element Configuration Model



Oracle Unified Directory treats all the participating elements equally; however, you must configure one participant as primary. You are not required to define a Join rule for a primary participant. In this figure, **P1** is the primary participant and all other participants **P2** though **P8**, are secondary participants.

Each secondary participant has a Join rule and has a *Joiner type*, which defines its relationship with another participant. For example, in case of P2, the Join rule `P2.cn=P1.cn` defines its relationship with P1 and if the Joiner type configured in P2 is `many-to-one`, then it implies that the relationship from P1 through P2 is `one-to-many`.

Participants P2, P8, and P5 are directly related to the primary participant P1, while the other secondary participants are indirectly related to the primary participant.

> **Note:**
>
> For more information about participants, Join rules, and Joiner types, see the following:
>
> - Understanding Join Participants
> - Overview of Join Rules
> - Understanding Supported Joiner Types

12.5.1.2 Understanding Join Participants

A Join *participant* is a workflow element that contributes some information to the Join workflow element to form a combined joined entry.

A Join workflow element can have one or more participating data sources, with each exposed through a workflow element. Participating workflow elements include:

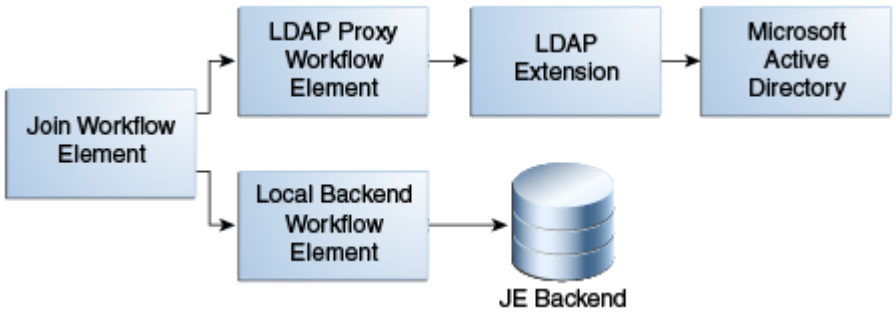
- Distribution workflow element
- Proxy LDAP workflow element
- Local Backend workflow element

- Load Balancing workflow element
- Another Join workflow element

For example, for each directory, you must create a Proxy LDAP workflow element that is associated with a directory to retrieve information from that directory. Afterward, you formulate these workflow elements as participants of a Join workflow element.

Figure 12-15 depicts the relationship between a Join workflow element and the participating workflow elements.

Figure 12-15 Join Workflow Element and Join Participants



A Join workflow element has only one *primary participant*, whose Directory Information Tree (DIT) structure is exposed by default, and can have one or more *secondary participants*. You determine which participant is primary.

You use the primary participant to create and search the directory tree entries. Entries must exist in the primary participant to be returned from a Join workflow element.

The Join workflow element takes each entry found in the primary participant and joins it with entries in other participants, based on the defined *Join rule*. You can also configure the Join workflow element to expose entries in the primary participant *and* entries that reside only in the secondary participants. For information about Join Rules, see Overview of Join Rules.

The Join workflow element and each participant must have an associated suffix (a DN).

- A Join workflow element DN is the virtual DN that is exposed through the workflow associated with that Join workflow element. You can configure the Join workflow element to restrict the view to only the Directory Information Tree that is of interest to the client.
- Ideally, a participant DN is the back-end naming context that is exposed through that participating workflow element or a descendant DN of that workflow element.

12.5.1.3 Overview of Join Rules

The following sections briefly describe the Join rules:

- Understanding Join Rules
- Understanding Attribute-Based Join Rules
- Understanding DN Join Rules

12.5.1.3.1 Understanding Join Rules

Join rules determine how an entry from one participant relates to an entry from another participant. Defining Join rules enables the Join workflow element to query secondary participants during LDAP operations.

> **Note:**
>
> - A Join rule always specifies the relationship between two, *and only two*, participants.
> - You define Join rules only for secondary participants and not for a primary participant.

The Join workflow element forms a search filter to search each secondary participant based on the Join rule defined for that secondary participant.

When you configure a Join workflow element, you must configure a Join rule for each secondary participant that specifies a relationship between entries in one participant with entries in the other participant. Also, the Join rule specified in at least one of the secondary participants must involve the primary participant, so that the Join workflow element can traverse the entire relationship tree starting from the primary participant.

Join rules identify the attributes of an entry from one participant to search another participant for obtaining the matching entries. These matching entries are then joined with the original entry to form the joined entry. When a matching value is found in the destination view, a join between the two entries is created.

The Join workflow element adds an attribute value, `joinedentrydn`, for each entry retrieved from a participating element. This value indicates which entries from secondary participants were used to form the consolidated entry. You can decide whether to configure the Join workflow element to populate this attribute, which might be useful when troubleshooting Join issues.

Oracle Unified Directory supports the following Join rule types:

- LDAP filter Join rules
- DN Join rules

Join rules follow LDAP filter syntax, which enables you to create complex Join rules using `AND` and `OR`. For example:

```
(&(P1.userId = P2.uid)(|(P1.deptNumber = P2.department)(P1.empNum = P2.empId)))
```

> **Note:**
>
> In a Shadow Join relationship, the Join rule must use the same attribute in both the primary and the shadow participant. For example, `p1.cn = p2.cn`.
>
> For more information, see Shadow Joiner Type.

Following are examples of valid Join rules:

- `p3.uid=p2.uid`
- `(&(P5.title=Primary.title)(Primary.cn=P5.sn))`

- `P5.dn = P7.dn`
- `P8.member = Primary.dn`
- `Primary.dn = P2.uniquemember`

> **Note:**
>
> the order in which you define the Join rule does not matter. For example, `P1.cn=P2.commonname` is the same as `P2.commonname=P1.cn`.

12.5.1.3.2 Understanding Attribute-Based Join Rules

Attribute-based Join rules define a Join relationship between two participants based on the common attribute values present in the matching entries from two participants.

For example, consider the Join rule `p1.uid=p2.username`, where p1 and p2 are two Join participants. This Join rule indicates that for an entry in p1, a corresponding matching entry from p2 is retrieved and joined with the entry from p1, if the `uid` attribute value of an entry from p1 matches the `username` attribute value of an entry from p2. If `uid` is a multi-valued attribute in p1, then the corresponding entry in p2 must match at least one of the values of p1. For instance, if the entry in p1 contains `uid=user.12` and `uid=user.34`, then the entry from p2 must contain either uid=user.12 or `uid=user.34`.

12.5.1.3.3 Understanding DN Join Rules

In some situations, the participating data sources do not have any attribute values in common except the entry DN. In these cases, you can configure a Join rule involving Entry DN.

A DN Join rule uses DN syntax and can take one of the following forms:

- The entry DN in one participant is constructed from an attribute in another participant. The DN must not contain the baseDN of the secondary participant, which makes it a relative DN. For example, you can configure the following DN join rule, which stipulates that the entry DNs in participant P2 must include the cn from participant P1, plus the ou=people suffix.

  ```
  P2.dn = "cn={P1.cn},ou=people"
  ```

- The entry DN in one participant matches an attribute in another participant. For example, you can configure this rule using the following syntax

  ```
  P8.member = P7.dn
  ```

  The preceding DN Join rule stipulates that the member attribute value in P8 should be used for locating the matching entries from P7.

- The entry DN in one participant is same as the entry DN in another participant. For example, you can configure this rule using the following syntax:

  ```
  P2.dn = P3.dn
  ```

  This Join rule stipulates that an entry DN in P2 must match an entry DN in P3 to form a joined entry. In this case, the Join rule looks for matching entries in portions of the DNs below the participant suffixes, although the full DNs may differ. For example, if participant P2 has a `dc=primary` suffix and participant P3 has a `dc=secondary` suffix, then the Join

rule implies that the trees below the suffixes are identical and it associates the "uid=user.1,cn=users,dc=secondary" entry with "uid=user.1,cn=users,dc=primary."

12.5.1.4 Overview of Join Policies

The following sections describe the different Join policies that govern joins between primary and secondary participants:

- Understanding Join Policies
- Example of Using a Join Policy

12.5.1.4.1 Understanding Join Policies

This section describes the different Join policies that govern joins between primary and secondary participants. Specifically, these policies determine which entries to return, including entries from only the primary participant, only from the secondary participant, or from both primary and secondary participants.

Oracle Unified Directory supports the following Join policy types:

- **Standard Join Policy Type:**

  If you specify the Standard Join policy type, then Oracle Unified Directory returns all entries in the primary participant that satisfy the search filter after joining the corresponding entries in the secondary participants.

- **Left Outer Join Policy Type**:

  If you specify the Left Outer Join policy type, then Oracle Unified Directory returns all entries in the primary participant after joining them with corresponding entries in secondary participants (by using a Standard Join), and then returns entries from the secondary participants that satisfy the join criteria and have a corresponding match in the primary participant. This process is equivalent to a Left Outer Join in database terminology.

  If you are joining entries from a secondary participant to a primary participant, then the join relationship is reversed. For example, a one-to-many join from the primary participant to a secondary participant becomes a many-to-one join, which is the same as a one-to-one join, from a secondary participant to the primary participant. Similarly a many-to-one join from primary to secondary becomes a one-to-many join from secondary to primary.

- **Full Outer Join Policy Type:**

  If you specify the Full Outer Join policy type, then Oracle Unified Directory returns all the entries in the primary participant after joining them with corresponding entries in secondary participants (by using a Standard Join), and then returns entries from the secondary participants that satisfy the join criteria and have a corresponding match in the primary participant (by using a Left Outer Join), and then returns entries from the secondary participants that satisfy the search filter, but do not have a matching entry in the primary participant. This process is equivalent to a Left Outer Join + Right Outer Join in database terminology.

  If you are joining entries from a secondary participant to a primary participant, then the join relationship is reversed. For example, a one-to-many join from the primary participant to a secondary participant becomes a many-to-one join, which is the same as a one-to-one join, from a secondary participant to the primary participant. Similarly a many-to-one join from primary to secondary becomes a one-to-many join from secondary to primary. For a Full Outer join, Oracle Unified Directory ignores the join condition for entries from secondary participants because it cannot compute the reverse of the join condition.

> **Note:**
>
> If you do not specify a particular Join type, then Oracle Unified Directory performs the Standard Join by default.

12.5.1.4.2 Example of Using a Join Policy

The following table illustrates how each of the Join Policies work. For this example, assume the following data resides in the primary participant and a secondary participant:

- The primary participant namespace is `dc=internal, dc=com`
- The secondary participant namespace is `dc=external, dc=com`
- The Join workflow element suffix is `dc=example,dc=com`

Table 12-2 How Join Policies Work

Data in Primary Participant	Data in Secondary Participant
dn: cn=Ronald, dc=internal,dc=com objectclass: inetorgperson cn: Ronald sn: Anne givenname: Anne Ronald telephonenumber: 54300	dn: cn=Ronald, dc=external,dc=com objectclass: inetorgperson cn: Ronald sn: Anne title: Manager
dn: cn=Sam, dc=internal,dc=com objectclass: inetorgperson cn: Sam sn: Ketty manager: cn=Ronald, dc=internal,dc=com telephonenumber: 54301	dn: cn=Sam, dc=external,dc=com objectclass: inetorgperson cn: Sam sn: Ketty title: SMTS
dn: cn=Richard,dc=internal,dc=com objectclass: inetorgperson cn: Richard sn: Rod title: Trainee manager: cn=Ronald, dc=external,dc=com telephonenumber: 54303 description: Trainee for dept 543 departmentNumber: 543	dn: cn=Richard,dc=external,dc=com objectclass: inetorgperson cn: Richard sn: Rod title: Trainee
dn: cn=William,dc=internal,dc=com objectclass: inetorgperson cn: William sn: Tent description: User with no title	dn: cn=Mike,dc=external,dc=com objectclass: inetorgperson cn: Mike sn: Ret title: MTS - dept_sec

12.5.1.5 Understanding Supported Joiner Types

A Joiner type defines the Join relationship between two participants. A Join relationship defines the way two Join participants are connected. In addition, a Join relationship between two participants is directed and defines the way a start participant is connected to the end participant. These Joiner types work for any kind of Join rule defined, complex or simple.

> **Note:**
>
> When a Join relationship from P1 to P2 with *many-to-one* Joiner type is configured, then internally Join workflow element implicitly creates a reverse relationship from P2 to P1 with *one-to-many* Joiner type and vice-versa. For a one-to-one Joiner and a shadow Joiner, the reverse relationship also contains the same Joiner type as that of the original relationship configured.

The following is a description of the supported Joiner types, including:

- One-to-One Joiner Type
- One-To-Many Joiner Type
- Many-To-One Joiner Type
- Shadow Joiner Type

12.5.1.5.1 One-to-One Joiner Type

The one-to-one Joiner, or simple join, defines a one-to-one relationship between the entries in two participants. In a one-to-one Joiner type, each entry in the start participant corresponds with one entry in the end participant of this relationship. If more than one matching entry exists in the end participant, then the Join workflow element uses the first returned entry from the end participant for the Join.

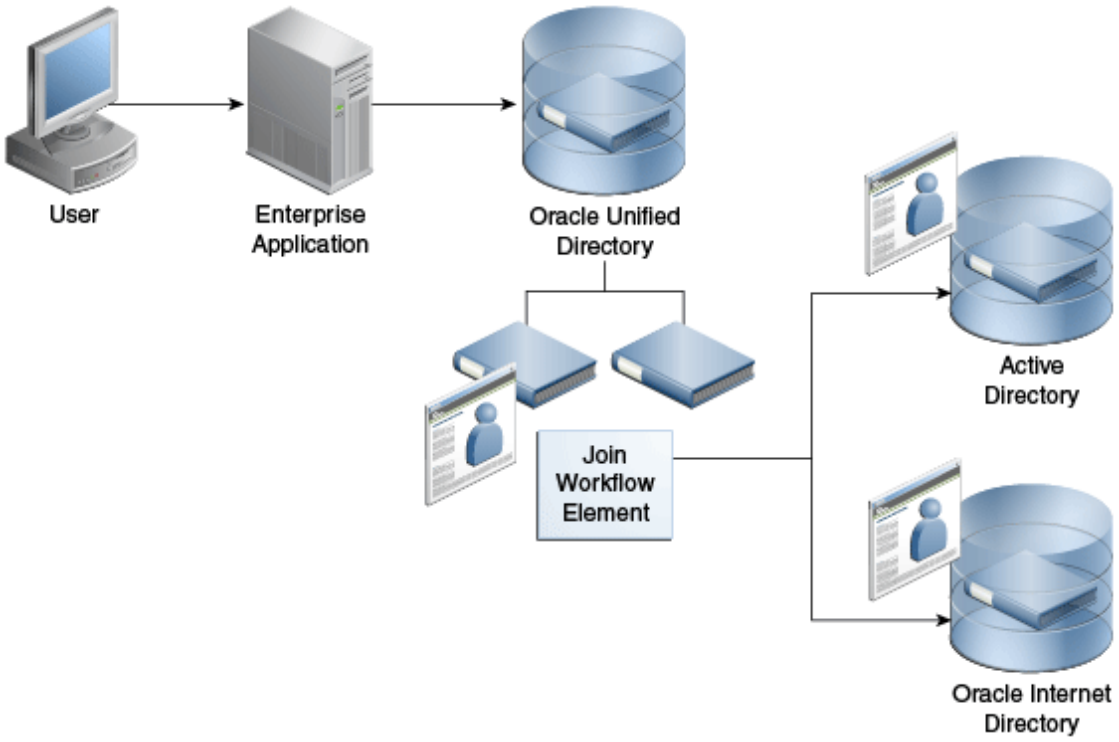
You can specify a more complex Join criterion involving a combination of `AND` and `OR` conditions using the LDAP filter syntax for the Join criteria. For example:

```
( & (P1.userId = P2.uid) ( | (P1.deptNumber = P2.department) (P1.empNum = P2.empId) ) )
```

In the preceding scenario, the search filter used for the secondary participant is coined based on the complex Join criteria configured. If the entry from primary participant does not have all the primary attributes specified in the Join rule, then the Join is not formed.

Figure 12-16 shows a a high-level example of a one-to-one Joiner used for authentication.

Figure 12-16 Sample One-to-One Joiner Type for Authentication



12.5.1.5.2 One-To-Many Joiner Type

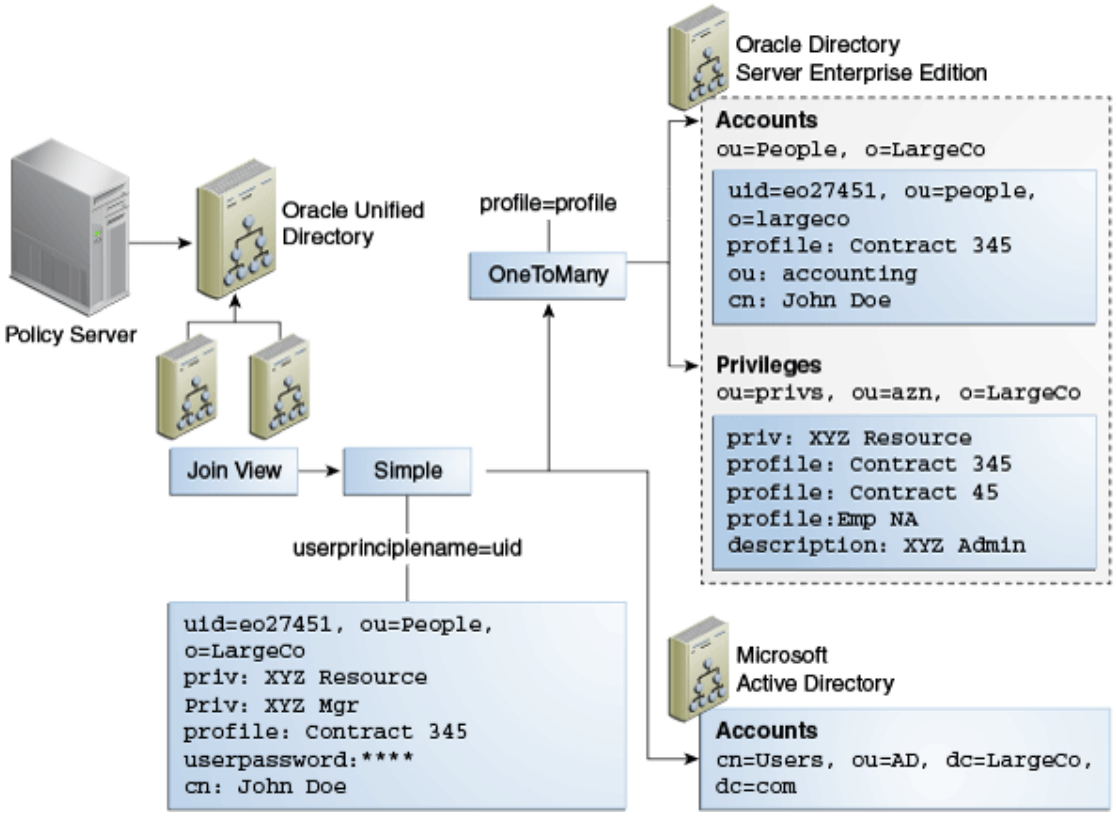
The one-to-many Joiner type defines a one-to-many relationship between two participants. Similar to a one-to-one Join relationship, the one-to-many Joiner locates entries in the end participant by comparing attributes; however, if an entry in the start participant corresponds with more than one entry in the end participant, this Joiner type consolidates all of the matching entries into one virtual joined entry.

The one-to-many Join is useful if you must consolidate multiple role objects or identities into one virtual entry.

Figure 12-17 depicts a scenario where a policy server makes policy decisions about an individual. For integration purposes, the policy server prefers to see a single entry with the rights of the user exposed as a privilege attribute, which allows the policy server to test rights assertions with queries such as:

```
ldapsearch -b "uid=e027451,ou=People,o=LargeCo" -s base "(priv=XYZ Mgr)"
```

Figure 12-17 One-To-Many Joiner Type



The one-to-many Joiner is used to match one or more privileges to a user, based on a profile attribute in their main `ou=People` entry. The one-to-many Joiner looks for all privileges with the same profile value as in the entry and merges them with the entry. A second stage Join uses the one-to-one so that the Oracle Directory Server Enterprise Edition (ODSEE) combined profile is used with the user's Active Directory credentials.

12.5.1.5.3 Many-To-One Joiner Type

The many-to-one Join relationship defines a many-to-one relationship between two participants, where multiple entries in the start participant have a corresponding single entry in the end participant. It is the inverse of one-to-many Joiner type.

For example, assume the primary participant contains a list of employee information and the secondary participant contains a list of department information. If multiple employees belong to one department, then a single department number in the secondary participant might apply to more than one employee in the primary participant.

However, if you delete an employee from the primary participant, you do not want to delete that employee's department number from the secondary participant. You can prevent this "cascading delete," by configuring a many-to-one relationship in the secondary participant. This relationship means that deleting an entry in the primary participant does not result in the deletion of the shared entry in the secondary participant.

12.5.1.5.4 Shadow Joiner Type

You sometimes need to store entries in a source, such as an LDAP store or a Database store, that requires a schema extension, but a schema extension is not possible either for business or technical reasons. The Shadow Joiner allows you to store the extended attributes in another store, such as Local Backend workflow element.

The Shadow Join relationship maintains the same structure of the entry in the primary participant, but stores additional attributes by creating shadow entries using a separate source. Using the Shadow Join relationship, applications can use the enterprise directory and also store application-specific attributes in the shadow directory such as Local Backend workflow element. The application believes it is communicating with a directory that stores all attributes, but Oracle Unified Directory silently stores application-specific data in an alternate *shadow* directory.

The Shadow Joiner encodes all primary participant DN's into a hash that is used to locate the matching entry in the shadow participant. If the Join workflow element fails to locate a corresponding record in the shadow participant, then it automatically creates a new one, storing the designated attributes in the shadow participant. The Shadow Joiner type operates transparently to the application, taking care of creating and renaming entries synchronized with that of the primary workflow element.

The Shadow Joiner supports all LDAP operations. When an LDAP modify operation occurs, the Shadow Join examines the parameters identified by the shadow participant's storable attributes to see if any of those attributes should be stored in the secondary participant. If any of these attributes exist, then the Shadow Join attempts to locate the local entry using the hash of the primary entry.

- If the Shadow Join locates the local entry, then it performs the appropriate LDAP modify operation on that entry.
- If Shadow Join does not find a local entry, it attempts a secondary search. The Shadow Join searches using a primary key, in case the primary DN changed.

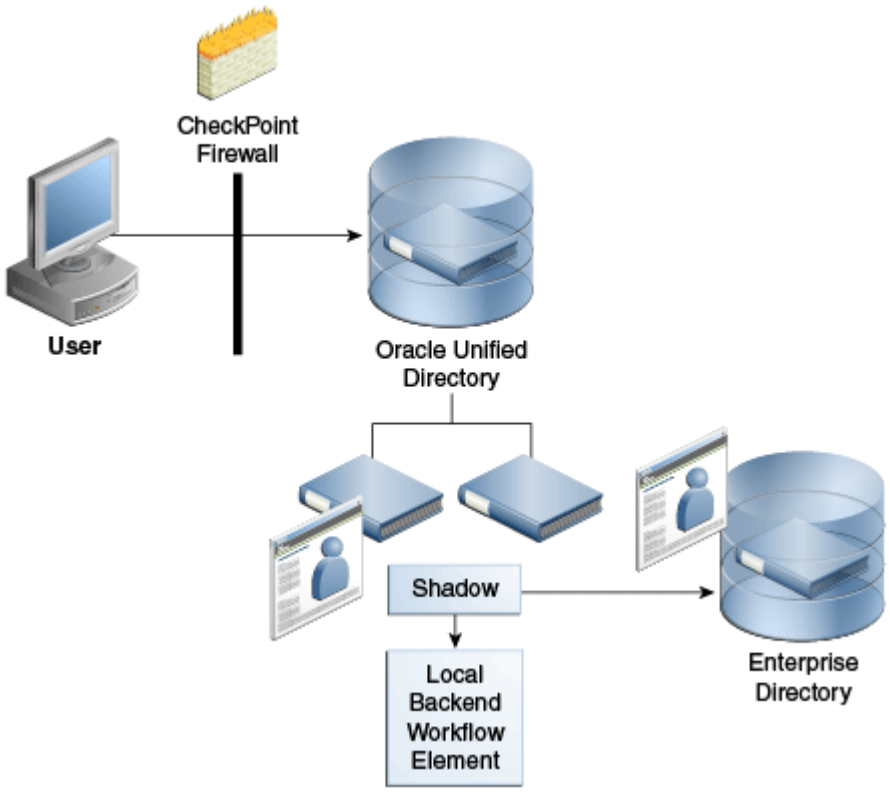
  If the local entry is still not found, the Shadow Join automatically creates a new entry.

> **Note:**
>
> For Shadow Joiner, the Join rule should involve the same attribute in both primary and shadow participant. For example, `p1.cn = p2.cn`.

You must ensure replication is configured for shadow back end to achieve high availability.

Figure 12-18 shows a firewall, for example CheckPoint, configured to connect to an Oracle Unified Directory. The Oracle Unified Directory uses Local Backend Database to maintain the firewall schema, allowing integration of the firewall into the corporate enterprise directory without requiring that the corporate enterprise directory schema be extended with application-specific data. Instead, by storing it in Oracle Unified Directory Local Backend database, the application-specific data can be managed by the team responsible for the firewall management.

Figure 12-18 Example Shadow Join Used for Storing Application-Specific Data Locally



12.5.1.6 Understanding the Join Condition

Oracle Unified Directory enables you to define a filter *condition* for a Join rule, where only those entries that satisfy the specified condition are considered for the Join. All entries that do not satisfy the condition are returned, as is, without a Join.

You can configure a join filter condition with any of the Joiner types described in this chapter. See Understanding Supported Joiner Types for a description of the different Joiner types.

Oracle Unified Directory always evaluates the Join condition with respect to the participant in which it is defined. In most situations, it is useful to define this Join condition only in the primary participant and not in other participants.

You specify a Join condition in the LDAP filter syntax, and you can define a Join condition using any complex filter using `OR` and `AND`. For example:

```
(&(employeenumber=101)(sn=Smith))
```

Oracle Unified Directory always evaluates the Join condition based on the participant in which it is defined. In the following example, Oracle Unified Directory considers only the users in P2 whose `sn` is *`Smith`* and `departmentNumber` is *`101`* for a join with P3, based on the `UserPrincipleName` attribute. So, if you defined this configuration for P2, then it is associated with participant P2.

```
ds-cfg-join-criteria: P2.uid = P3.UserPrincipleName
ds-cfg-join-condition: (&(departmentNumber=101)(sn=Smith))
```

12.5.1.7 About Virtual Attributes Creation

You can create virtual attributes based on the physical attributes stored in multiple participants of a Join workflow element. Because an attribute can be obtained from more than one participant, variables that define the attribute content must be fully qualified. That is, the source attribute value must include the name of the participant from which the attribute is taken.

For example, the following parameter creates a "mail" attribute from existing attributes in P1 and P2. This mail attribute is specified in the `ds-cfg-create-virtual-attribute` configuration parameter of the Join workflow element.

```
ds-cfg-create-virtual-attribute: mail =
 {P1.firstName}.{P2.lastName}@{P1.domainName}
```

In this case, the `firstName` and `domainName` attributes are taken from the P1 participant, and the `lastName` attribute is taken from P2.

The Join workflow element supports creation of virtual attributes based on individual attribute values from each participant. It also supports the simple concatenation or literal/attribute value assignment.

- `department = "ST"`
- `empid = P4.uid`
- `memberof = P8.dn`
- `mail = P3.CN + "." + P2.sn + "@oracle.com";`

12.5.1.8 Overview of Attribute Flow Settings

The following topics give an overview of the attribute flow settings, explains the working with retrievable, non-retrievable, storable and non-storable attribute settings:

- Understanding Attribute Flow Settings
- About Retrievable and Non-Retrievable Attribute Settings
- About Storable and Non-Storable Attribute Settings

12.5.1.8.1 Understanding Attribute Flow Settings

Each participating data source has the privilege to specify which attributes can be retrieved from it and which attributes can be stored in it. You can configure this privilege by specifying the following attribute flow settings for each participating workflow element:

- `retrievable-attribute` and `non-retrievable-attribute`
- `storable-attribute` and `non-storable-attribute`

Specifically, these settings control how attributes flow into and out of a Join participant. They also enhance security by preventing information from being requested by, or returned to, an unauthorized client. In addition, for Join workflow elements, the attribute flow settings control which attributes flow to which participant because multiple Join participants can contribute to the same virtual joined entry.

> **Note:**
>
> Unlike access controls, the attribute flow settings provide *silent enforcement*, which means they filter the request without returning an error to the client. In a high security environment, this silent enforcement prevents the client from knowing whether they are even allowed to see a particular attribute.

12.5.1.8.2 About Retrievable and Non-Retrievable Attribute Settings

When configuring a Join participant, both the retrievable-attribute and non-retrievable-attribute lists are empty by default, which means all attributes are retrievable. However, you can specify a list of attributes that the Join participant can or cannot retrieve as follows:

- Use the `retrievable-attribute` setting to specify a list of attributes that the Join participant can retrieve from the target directory.

  This setting contributes to server performance and, in some cases, security because you can only retrieve the specified attributes from a proxied server during `SEARCH` and `COMPARE` operations.

  In addition, you can use the retrievable-attribute setting to control attribute flow when using the Join workflow element. Because the Join workflow element Joins entries from multiple participants, you must configure the retrievable-attribute setting on each participant in the Join workflow element to restrict the flow of attributes from the participants in the Join view.
- Use the `non-retrievable-attribute` setting to specify a list of attributes that the Join participant cannot retrieve from the target directory.

Specifying a list of retrievable attributes indicates that only specific attributes may be requested from the proxied directory. An empty retrievable-attribute list indicates that all attributes are retrievable — unless you specify a list of non-retrievable attributes.

For example, you can retrieve attribute `A1` in the following circumstances:

- If both the retrievable-attribute and the non-retrievable-attribute lists are empty.
- If the retrievable-attribute list is empty, and the non-retrievable-attribute list does not contain `A1`.
- If the retrievable-attribute list contains `A1`, and the non-retrievable-attribute list does not contain `A1`.

12.5.1.8.3 About Storable and Non-Storable Attribute Settings

When configuring a Join participant, both the `storable-attribute` and `non-storable-attribute` lists are empty by default, which means all attributes are storable. However, you can specify a list of attributes that the Join participant can or cannot store as follows:

- Use the `storable-attribute` setting to specify a list of attributes that the Join participant can store on the target directory.

  This setting contributes to server performance and, in some cases, security because only the specified attributes and their values are sent to the proxied server for ADD and MODIFY operations.

  In addition, you can use the storable-attribute setting to control attribute flow when using the Join workflow element. Because the Join workflow element Joins entries from multiple

participants, you must configure the storable-attribute settings on each participant in the Join view to restrict the flow of attributes from participants in the Join view.

- Use the `non-storable-attribute` setting to specify a list of attributes that the Join participant cannot store on the target directory.

Specifying a list of attributes indicates that only specific attributes can go to the participating workflow element. An empty `storable-attribute` list indicates that all attributes are storable — unless you specify a list of non-storable-attributes.

For example, you can store an attribute in the following circumstances:

- If both the `storable-attribute` and `non-storable-attribute` lists are empty.
- If the `storable-attribute` list is empty, but the `non-storable-attribute` list does not contain that attribute.
- If the storable-attribute list contains the attribute, but the `non-storable-attribute` list does not contain that attribute.

12.5.1.9 About Bind Operations

The Join workflow element supports user authentication for all participants. The Join workflow element provides a bind fall-through feature that allows you to try password validation against more than one data source. You must authenticate users against more than one data source because user identities might exist in multiple directories and passwords might be stored in any of the data sources.

To use the bind feature, you must configure the bind as an enabled operation in that participant. Use the `--set participant-bind-priority` configuration parameter to assign a bind priority to each participant in the Join workflow element, which determines the participant's priority in processing the bind.

Each participant is assigned a bind priority and the bind falls through all of the bind participants in the specified order until a successful bind is achieved. A bind failure is returned only when all bind participants have failed to authenticate the user.

The bind priority can be any positive integer that is greater than or equal to zero. The priority decreases from zero to higher integers. That is, the participant with least number has the highest bind priority, the participant with the next least number has the next higher bind priority, and so on. Zero has the highest priority.

If there is only one bind participant, then the bind error message from that participant is returned if the bind fails. However, if there are multiple bind participants, then the bind error message from the primary participant is returned if the bind fails for all the participants.

12.5.1.10 About DN Attributes Translation

DN attributes is a list of attributes to be treated as DNs for which namespace translation is required, such as `member`, uniquemember, and `manager`. For example, when reading a group entry from a proxied directory, Join workflow element converts the DN for the group entry and the `uniquemember` or `member` attributes if these attributes are in the DN attributes list.

> **Note:**
>
> Translate only those attributes that are needed by the client application. Entering all possible DN attributes may not be necessary for an application.

12.5.1.11 Configuring the Criticality of Join Participants

The `criticality` configuration parameter determines how the Join workflow element behaves when a search being performed against a participant fails due to a host error. Criticality applies only to search requests.

- `WRITE` operations are always critical in all participants.
- `BIND` and `COMPARE` operations are always non-critical in all participants, so that they can fall-through all eligible participants until a success is found.

To configure criticality for a Join participant, use the `dsconfig set-join-participant-prop` subcommand and set one of the following criticality flag values:

- `true` (*default setting*)

  This setting indicates that the participant is considered critical. If a participant fails to return a result because of an operation error, then the Join workflow element causes the overall search operation to fail and returns a `DSA Unavailable` error to the client, regardless of whether data was found in any other participant or not.

- `false`

  This setting indicates to the Join workflow element that the failure to perform an operation in the participant is not critical to the overall result. If a non-critical participant incurs an operations error, then that result is omitted from the overall LDAP search results. The Join workflow element returns partial results from any other participant and does not indicate any error.

- `partial`

  This setting indicates that the participant is partially critical, which implies that the application can notify its own users that partial results were obtained. If a partially-critical participant fails to return a result because of an operation error, then the Join workflow element returns not only partial results but also an `Admin Limit Exceeded` error. While this is not the expected error, the intention of this setting is to cause client application logic to indicate that not all results are shown.

For example, the following command sets the criticality of a participant named `joinparticipant-1` to `true`:

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file \ set-join-
participant-prop --element-name we-join \
--participant-name joinparticipant-1 \
--set participant-criticality:true
```

12.5.1.12 Understanding Enabled Operations

You use the `ds-cfg-enabled-operation` parameter to configure which LDAP operations to perform on a Join participant. These operations include:

- `ADD`
- `BIND`
- `COMPARE`
- `DELETE`
- `MODIFY`
- `MODIFYDN`

- SEARCH

The participant can participate only in the operations that you specify in this parameter.

- The COMPARE, DELETE, MODIFY, and SEARCH operations are enabled by default.
- You must configure BIND as an enabled operation to allow a participant to participate in bind operations. If you enable the BIND operation, then the configured bind priority determines the participant's priority when the bind is processing.
- The ADD and MODIFYDN operations are not enabled by default.
  - If the Join is a shadow join, you can enable ADD and MODIFYDN on primary and secondary participants.
  - If the Join is not a shadow join, you can only enable ADD and MODIFYDN on the primary participant.

12.5.1.13 Understanding How to Cascade Write Operations to Secondary Participants

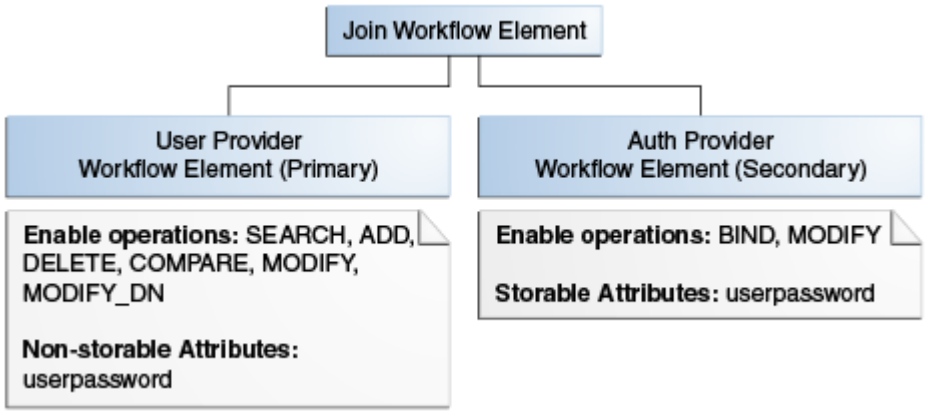
You can cascade write operations, such as DELETE and MODIFY, to secondary participants. That is, if you delete an entry a primary participant, then the related entries in all secondary participants are also deleted. However, this cascading operation is applicable only when you configure the DELETE operation as an enabled operation in the associated secondary participant and the relationship between the original participant and the to-be-cascaded-delete participant is not many-to-one.

MODIFY operations are also cascaded to all eligible secondary participants based on their enabled-operations configuration and storable attribute configuration. ADD and MODIFYDN are cascaded only to shadow secondary participants based on the storable attributes configured in those participants.

12.5.1.14 Understanding How to Use the Join Workflow Element to Implement Pass-Through Authentication

You can configure the Join workflow element to delegate bind requests to the Authentication (Auth) Provider workflow element and non-bind requests to the User Provider workflow element, as depicted in Figure 12-19. This configuration also takes care of delegating MODIFY PASSWORD requests to the Auth Provider workflow element and other MODIFY operations to the User Provider workflow element.

Figure 12-19 Pass-Through Authentication Using the Join Workflow Element



> **Note:**
>
> For a simple pass-through authentication scenario, use the pass-through authentication workflow element described in Overview of the Pass-Through Authentication Mechanism.

Use the Join workflow element to configure pass-through authentication only if you have special requirements that cannot be met by using the pass-through authentication workflow element. For example, the pass-through authentication workflow element does not route bind requests to the User Provider workflow element, and the user password is stored only in the Authentication Provider workflow element. If you want a different deployment scenario, where you want to store the user password in both the Authentication Provider workflow element and the User Provider workflow element, then you can use the Join workflow element and configure both providers to handle bind requests.

12.5.1.15 Handling LDAP Operations in Join Workflow Elements

If an attribute exists in both the primary and secondary participants, then the Join workflow element merges the attribute values. For read operations, this implies that a multi-valued attribute is returned with the values from all participants. For write operations, the proxy queries all participants and determines where to write the value based on the storable attributes configured in each Join participant.

This section contains the following topics:

- Considerations for Configuring the Join Workflow Elements
- Processing LDAP Operations Using the Join Workflow Element

12.5.1.15.1 Considerations for Configuring the Join Workflow Elements

When configuring the Join workflow element, you must keep the following points in mind:

- When you have multiple attributes with the same name from multiple data sources, such as two `uid` attributes from two different Proxy LDAP workflow elements, the Join workflow element only displays a single value.

  However, you can configure the Join workflow element to retrieve attribute values from a specific participant. To do this, remove the attributes from the Retrievable Attributes field for any participants for whom you do not want to view the attribute.
- You must configure virtual ACIs correctly to grant or deny entries and attributes from a Join workflow element.
- When using a Proxy LDAP workflow element as a Join participant, the credentials you use to perform operations in each participant plays a significant role, as follows:
  - If you configure the `use-specific-identity` bind mode in the Proxy LDAP workflow element, then only a specific identity is used for all non-bind operations.
  - If you configure the `use-client identity` bind mode in Proxy LDAP workflow element, then actual client credentials are used when `userDN` is a descendant of any of the DNs configured in the `include-list` of a Proxy LDAP workflow element. Otherwise, Oracle Unified Directory uses a specific identity to perform operations in the Proxy LDAP workflow element.

- All Proxy LDAP workflow elements must set the `include-list` to the respective user container DNs so that the bind correctly happens either with the client DN or with a specific identity. This configuration also requires each participant's user container to be unique, or the bind fails.
- You must always configure the proxy and root credentials in the Proxy LDAP workflow element because some internal operations use those credentials. These credentials are also required when you configure a `include-list` in a Proxy LDAP workflow element.

12.5.1.15.2 Processing LDAP Operations Using the Join Workflow Element

Table 12-3 describes how the Join Workflow element processes each LDAP operation.

> **Note:**
>
> The Join workflow element displays a single value only, if there are multiple attributes with the same name from multiple data sources, for example two `uid` attributes from two different Proxy LDAP workflow elements. However, you can configure the Join workflow element to retrieve attribute values only from a specific participant. To view the attribute from a specific participant only, you must ensure that the attribute is not listed in the Retrievable Attributes field for the participant for which you do not want to show the attribute for.

Table 12-3 How the Join Workflow Element Processes LDAP Operations

LDAP Operation	Processing Description
`ADD`	• Processed in the primary participant based on storable attributes and enabled operation. • Processed only in shadow secondary participant based on storable attributes configured in that participant. • For other Joiners, no processing is done for secondaries. Store entry in the secondary participant if at least one attribute (except the link attributes) must be stored in the shadow. Typical shadow Join participant has storable attributes set. Implicitly add link attributes to storable. • Implicitly treat all Join attributes as storable, unless they are configured in the unstorable attributes list.
`BIND`	Processed in each bind participant based on bind priority.
`COMPARE`	• Processed in primary participant based on retrievable attributes and enabled operation. • If COMPARE failed in primary participant, then `COMPARE` is processed in all secondaries based on retrievable attributes and enabled operations.
`DELETE`	• Processed for all participants where `DELETE` is enabled. • Not processed for participants that are on `1` side of a many-to-one relationship.

Table 12-3 (Cont.) How the Join Workflow Element Processes LDAP Operations

LDAP Operation	Processing Description
MODIFY	• Processed in primary participant based on storable attributes and enabled operation. • For secondaries (any Joiner type), process modification if attributes are storable attributes. • For a Shadow Joiner Modify the shadow entry if the MODIFY attribute must be stored in the shadow participant. If the shadow entry is missing, then create a new entry to store the MODIFY attribute if the attribute must be stored in the shadow participant. If the shadow entry is retrieved by searching a second shadow, then rename the shadow entry to a correct value. • For all Joiner types, implicitly treat all attributes as unstorable to maintain the link. Does not allow modification of link attribute through the Join workflow element. The operation succeeds, but the link attribute is not modified. • In all participants where the MODIFY attribute is not a link attribute, the modification takes place if that attribute is defined as storable.
MODIFYDN	• Processed in primary participant based on enabled operation. • For Shadow Joiner, update the shadow entry DN. • For other Joiner types, no processing done for secondary participants. • For all Joiners, does not allow MODDN for link attributes and if deleteoldrdn is true.
SEARCH	Processed in primary participant first. Then, joins the entry with all eligible secondary participants for each entry returned from the primary that satisfies the Join condition.

12.5.2 Overview of Optimizing Search Results From Virtual Directories Using Workflow Elements

Oracle Unified Directory provides two workflow elements that automatically narrow search results to help you more efficiently view or retrieve data from virtual data sources.

You can insert the following workflow elements into any workflow that returns search results:

- GetRidofDuplicate

  The GetRidofDuplicate workflow element removes, from search results for the current search operation, all the entries whose DN has already been returned to the client application. This is useful when a workflow element is likely to return several entries, maybe hundreds, with the same DN.

- HideByFilter

  The HideByFilter workflow element enables you to control in fine detail which entries are returned by search operations. For example, if you use Oracle Unified Directory as an address book directory, you can display only the entries for customer service representatives. First, give customer service representatives an ou value such as CSR. Then you can use the HideByFilter workflow element with the hideByFilter set to ou=CSR. When the directory is searched, only the customer service representatives entries are returned.

For detailed information about configuring the GetRidOfDuplicates and HideByFilter workflow elements, see Optimizing Search Results From a Virtual Directory .

12.5.3 Understanding Addition of memberof User Attributes to person Entries

You can configure the VirtualMemberof workflow element to add the memberof user attribute to person entries, which is useful when you want applications see group membership, but do not want them to perform secondary searches for those groups.

The memberof attribute values are the DNs of any groups to which the person entry belongs.

> **Note:**
>
> The VirtualMemberOf workflow element only impacts the SEARCH operation
>
> For information about creating and configuring a VirtualMemberof workflow element, see Adding the memberof User Attribute to person Entries.

You can use the memberof attribute in a search filter (for example, MemberOf=group1); however, memberof does not support the following combinations:

- PRESENT, SUBSTRING, GREATER_OR_EQUAL, LESS_OR_EQUAL for memberof
- OR filter with an inner memberof component
- NOT filter with an inner memberof component
- Pattern-based search filter, such as a wildcard character (for example, memberOf=cn=*)

Based on these restrictions, an inner memberof component is supported only in AND filters.

When you use the memberof attribute in the search filter, Oracle Unified Directory only returns entries with objectclass=person. The VirtualMemberof workflow element does not support using the memberof attribute on a non-person entry.

12.5.4 Overview of Renaming DNs Using the Proxy

You can rename DNs using the proxy as explained in the following topics.

This section includes the following topics:

- About DN Renaming Using the Proxy
- Understanding How the DN Renaming Workflow Element Works

12.5.4.1 About DN Renaming Using the Proxy

Each entry in a directory is identified by a DN and a set of attributes and their values. Sometimes, the DN and the attributes defined on the client side do not map with the DN and the attributes defined on the server side. For instance, an organization, Example A contains dc=parentcompany, dc=com entries. It acquires another organization, Example B. Example B contains dc=newcompany, dc=com entries. Therefore, dc=newcompany, dc=com must be renamed to dc=parentcompany, dc=com for the existing client applications to work correctly.

You can define a DN renaming workflow element to rename DNs to values that match the server side. When a client makes a request, the DNs and attributes are renamed to match those in the server. When the result is returned to the client, the DN and attributes are changed back to match what the client has requested.

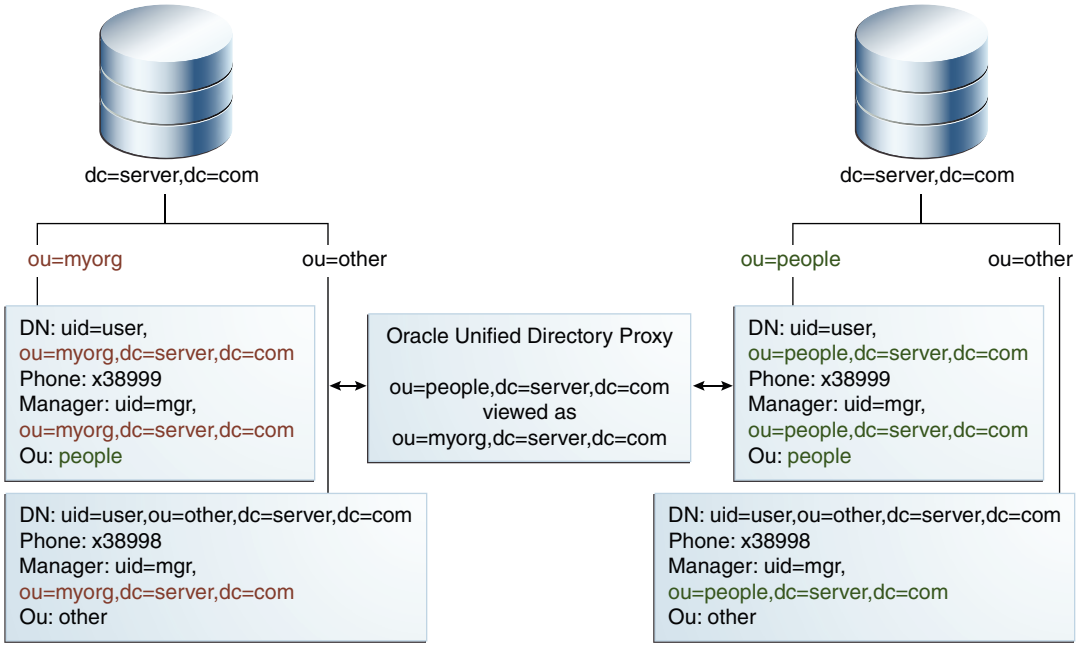
> **Note:**
>
> For information about configuring DN renaming, see Performing DN Renaming.

12.5.4.2 Understanding How the DN Renaming Workflow Element Works

Oracle Unified Directory provides a DN renaming workflow element that allows you to transform the content of a Directory Information Tree (DIT) into another DIT with a different base DN. When an operation (Add, Bind, Delete, Modify, and so on) goes through a DN renaming workflow element, its parameters are transformed according to the DN renaming configuration to transform the virtual entries into real entries.

Figure 12-20 illustrates how DN renaming is performed using the proxy.

Figure 12-20 DN Renaming



The client expects `ou=myorg`, `dc=server`, `dc=com` entries. However, the LDAP server contains `ou=people`, `dc=server`, `dc=com` entries. The proxy renames the DNs by making use of the DN renaming workflow element.

In this example, the real entries `ou=people`, `dc=server`, `dc=com` are seen as `ou=myorg`, `dc=server`, `dc=com` entries from the client side.

The DN renaming transformation is applicable to the following objects:

- DN of the entry

  For example, the real entry on the LDAP server `dn:uid=user`, `ou=people`, `dc=server`, `dc=com` is transformed into a virtual entry `dn:uid=user`, `ou=myorg`, `dc=server`, `dc=com` from the client perspective

- Attributes of the entry that contain either DNs or Name And Optional UIDs syntax

  For example, the server-side value of the `manager` attribute of an entry with an objectclass `inetorgperson` has a DN syntax: `manager: uid=mgr,ou=people,dc=server,dc=com` and is transformed into the value `manager: uid=mgr,ou=myorg,dc=server,dc=com` on the client side.

  In another example, the server-side value of the `uniquemember` attribute has a Name And Optional UID syntax (as defined in RFC 4517) as `uniquemember: uid=member,ou=people,dc=server,dc=com#'0111'B` and is transformed into the value `uniquemember: uid=member,ou=myorg,dc=server,dc=com#'0111'B` on the client side.

> **Note:**
>
> You can apply the transformation to all the user attributes of the entries, define a restricted list of attributes to which the operation applies, or define a restricted list of attributes to which the operation does not apply.

12.5.5 Understanding How to Modify RDN Values Using the Proxy

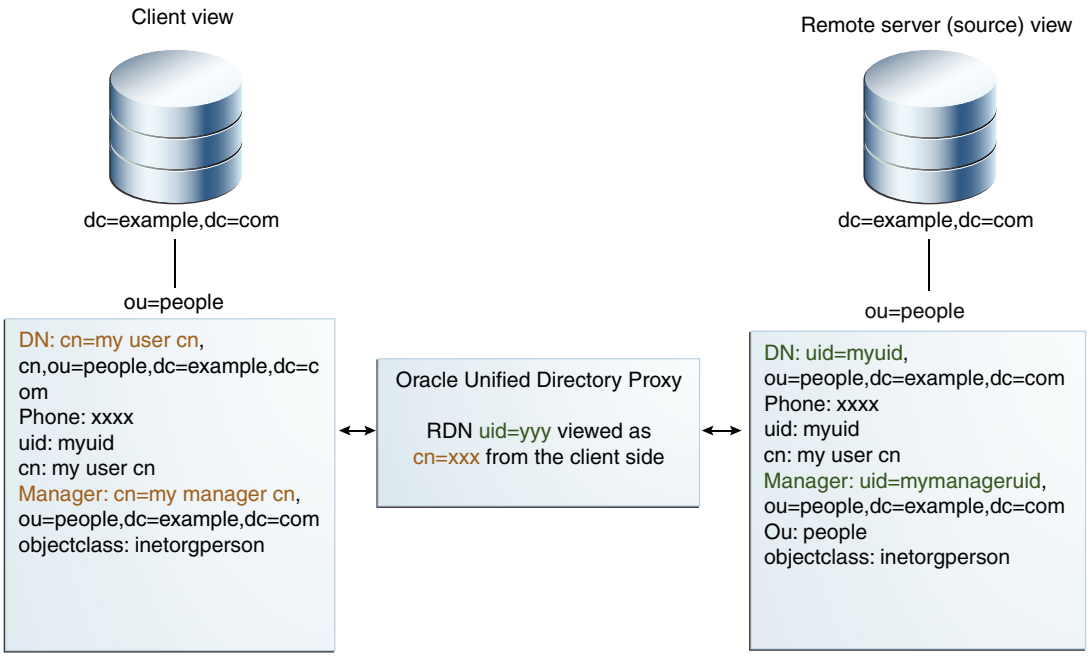
Oracle Unified Directory enables you to rename or replace RDN values from the source directory to Oracle Unified Directory using the RDNChanging configuration.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

> **Note:**
>
> For information about configuring RDN changing, see Performing RDN Changing Configuration.

Figure 12-21 illustrates how RDN changing is performed using the proxy.

Figure 12-21 RDN Changing



> **Note:**
>
> The relative distinguished name (RDN) is the leftmost element in an entry DN. For example, the RDN for `uid=Marcia Garza,ou=People,dc=example,dc=com` is `uid=Marcia Garza`. You can only change the leftmost element in an entry DN.

The RDNChanging configuration has the following parameters:

- objectclass

  Identifies the objectclass type that RDN renaming is performed on. The default setting is `person`.

- replace-value

  True or False: Indicates whether the value of original RDN value in the client view (identified by the `source-rdn` parameter) should be replaced by the value of the new RDN value (identified by the `client-rdn` parameter). The default setting is `true`.

  > **Note:**
  >
  > When the value is set to true, and an entry has multiple values for the new RDN attribute, then Oracle Unified Directory uses the first value in RDN.

- source-rdn

Identifies the original RDN attribute name from the source directory to be replaced or renamed in Oracle Unified Directory.

- client-rdn

  Identifies the new RDN attribute name to be used in Oracle Unified Directory and replaces the attribute name identified by the source-rdn configuration parameter.

- dn-attributes

  List of attributes with DNs to perform RDN renaming on. The default list of attributes are `member`, `manager`, and `owner`.

12.5.6 Understanding How to Retrieve Attributes from a SAML Identity Provider Using SAML XASP

Oracle Unified Directory provides a workflow element that retrieves attributes from a SAML Identity Provider using the SAML X.509 Attribute Sharing Profile.

- Overview of SAML XASP Workflow Element
- Configuration Parameters for SAML XASP Workflow Element

For more information on how to create a new SAML XASP workflow element or how to modify the properties of an existing workflow element, see Configuring SAML XASP.

12.5.6.1 Overview of SAML XASP Workflow Element

The SAML XASP workflow element allows Oracle Unified Directory to retrieve attributes from a SAML Identity Provider using the SAML X.509 Attribute Sharing Profile.

Oracle Unified Directory sends SAML V2.0 assertion queries to an identity provider. The content of the SAML response is then used to build an entry that will be returned by the SAML XASP workflow element.

The SAML XASP workflow element can operate on three types of search operations:

- A subtree search on the custom adapter base with a filter containing (*certificatedn=cn=user1,cn=users,dc=example,dc=com*)

  This search is transformed into a SAML V2.0 assertion query requesting the information for the subject *cn=user1,cn=users,dc=example,dc=com*. If the query returns an entry, this entry is added in the cache and indexed by its *cn*.

- A base search on the *cn=user1,cn=users,dc=example,dc=com* entry

  This search is transformed into a SAML V2.0 assertion query requesting the information for the subject *cn=user1,cn=users,dc=example,dc=com* or returns the entry if it is already contained in the cache.

- A subtree search on the custom adapter base with a filter containing (*cn=user1*)

  The entry is searched in the cache.

> **Note:**
>
> In the preceding descriptions, you can replace *certificatedn* with the value of the `xasp-attribute-name` configuration parameter, and *cn* with the value of the `xasp-index` configuration parameter.

12.5.6.2 Configuration Parameters for SAML XASP Workflow Element

Learn about the various SAML XASP workflow element configuration parameters, their names, descriptions, allowable values and formats, and rules to perform an action based on the values provided, from the tabular column below.

Table 12-4 describes the configuration parameters needed to configure SAML XASP workflow element:

Table 12-4 Configuration Parameters for SAML XASP Workflow Element

Property	Description	Mandatory or Optional	Value
`xasp-ws-url`	Defines the web service URL of the identity provider.	Mandatory	String For example, *http://hostname:port/fed/ar/soap*.
`xasp-attribute-name`	Defines the attribute that contains the DN of the entry in the search filter.	Mandatory	String For example, *certificatedn*.
`xasp-contains-dn`	Defines a string that must be part of the DN of the entry. For example, *dc=example*. If the DN of the entry does not contain this exact string (ignoring case), then the entry is not returned. This parameter allows the workflow element to restrict the searches to a portion of the DIT, for example only entries below *cn=users*. If you do not want to restrict the searches, specify *dc=example*.	Mandatory	DN For example, *dc=example*.
`xasp-base-dn`	Base DN to be used in case the value is not present in filter.	Mandatory	DN For example, *dc=example,dc=com*.
`xasp-ttl`	Defines the Cache Time-To-Live in milliseconds (Default value and minimal value is 5000).	Optional	String
`xasp-debug`	Specifies whether to log additional messages. • `true`: Log additional messages. • `false`: Do not log additional messages. (Default)	Optional	Boolean `true, false`
`xasp-reverse-dn`	Defines whether the DN contained in the certificate is reversed. For example, *dc=com,dc=example,cn=users,cn=user1*. • `true`: Specify if the dn is reversed. • `false`: Specify if the dn is not reversed. (Default)	Optional	Boolean `true, false`

Table 12-4 (Cont.) Configuration Parameters for SAML XASP Workflow Element

Property	Description	Mandatory or Optional	Value
`xasp-index`	Defines a comma-separated list of attributes that is used to index the results. **Note:** To be indexed, this attribute must be part of the SAML response.	Optional	String For example, *cn*.
`xasp-response`	Defines a comma-separated list of attribute value pairs that is added to each returned entry.	Optional	String For example, *department=Sales*.

12.5.6.3 Considerations for Using the SAML XASP Workflow Element

Consider the following points while using the SAML XASP workflow element:

- None of the logical operators have any significance in the LDAP search filter
- The LDAP search filter works only for `xasp-attribute-name` and `xasp-index` (if `xasp-attribute-name` is not present in the filter)
- Regular expressions do not work with the LDAP search filter for SAML XASP workflow

12.5.7 Understanding ForkJoin Workflow Element

The ForkJoin workflow element aggregates data from two remote data sources at real time.

This section describes the following topics:

- Overview of ForkJoin Workflow Element
- About ForkJoin Participants
- Configuration Parameters for ForkJoin Workflow Element

12.5.7.1 Overview of ForkJoin Workflow Element

Oracle Unified Directory supports ForkJoin workflow element. It solves the problem of split-profiles, where the user data is split across two repositories.

The ForkJoin workflow element can have one primary participant and one secondary participant, each of which is a workflow element of any type. The ForkJoin workflow element allows you to search (search filter) against the primary participant, the secondary participant, or both the primary and the secondary participant. However, the Join workflow element allows you to search (search filter) against the primary participant only.

For example, consider a scenario where user data resides in multiple identity sources, with `samaccountname`, `sn`, `givenname`, and `employeenumber` in Active Directory. However, the `title` attribute is stored in a Human Resources database. Assume, Active Directory configured as the primary participant and Human Resources database configured as the secondary participant. Now, with ForkJoin workflow element configured, when an LDAP-enabled application queries the user data based on `samaccountname`, or `title`, or both then Oracle Unified Directory retrieves user entries based on the filter from both Active Directory and the Human Resources database.

12.5.7.2 About ForkJoin Participants

ForkJoin participants are the workflow elements that are participating in the ForkJoin configuration topology. There are two types of ForkJoin participants, the primary participant and the secondary participant.

The ForkJoin workflow element can have one primary participant and one secondary participant, each of which is a workflow element of any type. You determine which participant is primary.

For each data backend (LDAP or RDBMS), you must create either a Proxy LDAP workflow element or RDBMS workflow element that is associated with the backend to retrieve information from that backend. You can then formulate these workflow elements as participants of a ForkJoin workflow element topology.

> **Note:**
>
> See Implementing ForkJoin Workflow Element Configuration Model to learn how to deploy a ForkJoin configuration.

You must keep the following points in mind while implementing ForkJoin workflow element configuration:

- ForkJoin workflow element configuration supports only a single secondary participant.
- ForkJoin workflow element configuration allows simple Join rule of the format `primary-participant-attribute=secondary-participant-attribute`. For instance, `sn=cn` where `sn` is the attribute in the primary participant and `cn` is the attribute in the secondary participant.
- When `join-policy` is set to `full-outer-join` and BIND operation is enabled on secondary participant, then bind is allowed even for secondary-only users that do not exist in the primary participant at all.

12.5.7.3 Configuration Parameters for ForkJoin Workflow Element

Learn about the various ForkJoin workflow element configuration parameters, their names, descriptions, and functionality.

The following table describes the ForkJoin workflow element configuration parameters:

Table 12-5 Configuration Parameters for ForkJoin WorkFlow Element

Property	Description	Mandatory or Optional	Value
`cache-size`	Defines the maximum number of entries that can be stored in the cache used to filter out entry duplicates. When the entry duplicates filtering is active, the ForkJoin workflow element stores the returned entry DNs in a cache. A cache is specific to a search operation, and when the cache is full, the search operation is aborted. By default, the cache can handle `10000` DNs. When the cache size is set to `0` or is negative, then no limit is enforced.	Optional	`10000`
`dn-attribute`	Defines the DN syntax attribute whose DN need to be translated to that of ForkJoin workflow element.	Optional	`manager`, `member`, `memberof`, `uniquemember`
`enabled`	Indicates whether the workflow element is enabled for use in the server. If a workflow element is not enabled, then its contents are not accessible when processing operations.	Mandatory	`true, false`
`join-suffix`	Defines the virtual DN that will be exposed by the ForkJoin workflow element.	Mandatory	A valid DN, for example `dc=example,dc=com`.
`populate-joinedentrydn`	Decides if the attribute `joinedentrydn` has to be populated in the joined entries.	Optional	`true, false`

Table 12-5 (Cont.) Configuration Parameters for ForkJoin WorkFlow Element

Property	Description	Mandatory or Optional	Value
`primary-and-secondary-attributes`	Defines a list of attributes that are present in both the primary and secondary participants and which the application can use in the search filter. The attributes identified by the `primary-and-secondary-attributes` configuration parameter cannot also be identified by the `secondary-only-attributes`configuration parameter. The `objectclass` attribute is always considered as `primary-and-secondary-attributes` by default, even if not explicitly configured.	Optional	The name of an attribute type defined in the server schema.
`secondary-only-attributes`	Defines a list of attributes that are present only in the secondary participant and which the application can use in the search filter. The attributes identified by the `secondary-only-attributes` configuration parameter cannot be identified by the `primary-and-secondary-attributes` configuration parameter.	Optional	By default, the `secondary-only-attributes` attributes list is empty, which means that participant is not responsible for handling any attributes that are present in search filter. When it is empty for all the participants then all search attributes are considered to be in primary participant. The value is the name of an attribute type defined in the server schema.

Table 12-5 (Cont.) Configuration Parameters for ForkJoin WorkFlow Element

Property	Description	Mandatory or Optional	Value
`join-policy`	Defines the Join policy configuration for this ForkJoin workflow element, which decides what entries are returned from search operation. This is analogous to Database join. It supports the following values: • `standard-join`: Returns all entries that satisfy the search filter in the primary participant after joining the corresponding entries in secondary participants. • `left-outer-join`: Returns all the entries in the primary participant after joining with corresponding entries in secondary participant (`standard-join`), and returns entries from the secondary participant that satisfy the join condition and which have a corresponding match in primary participant. This is the equivalent to Left Outer Join in database terminology. • `full-outer-join`: Returns all the entries in the primary participant after joining with corresponding entries in secondary participant (`standard-join`); returns entries from the secondary participant that satisfy the join condition and which	Mandatory	`standard-join`, `left-outer-join`, `full-outer-join`

Table 12-5 (Cont.) Configuration Parameters for ForkJoin WorkFlow Element

Property	Description	Mandatory or Optional	Value
	have a corresponding match in primary participant (`left-outer-join`); and returns entries from the secondary participant that satisfy the search filter but do not have a matching entry in primary participant. This is the equivalent to Left Outer Join + Right Outer Join in database terminology. For conditional joins when `full-outer-join` is configured, the entries in secondary participant are mapped to entries in primary participant without considering the filter in join condition.		

> **Note:**
>
> At a minimum, you must set either the `secondary-only-attributes` or `primary-and-secondary-attributes` configuration parameter to implement the ForkJoin workflow element. See Implementing ForkJoin Workflow Element Configuration Model to learn how to deploy a ForkJoin configuration.

12.5.8 Understanding DynamicGroups Workflow Element

The DynamicGroups Workflow Element operates by monitoring the returned LDAP objects and detects the entries where the `memberURL` attribute is present and the objectclass is `groupOfURLs`.

This section contains the following topics:

- Overview of DynamicGroup Workflow Element
- Configuration Parameters for DynamicGroup Workflow Element

12.5.8.1 Overview of DynamicGroups Workflow Element

The DynamicGroups Workflow Element enables Oracle Unified Directory to process LDAP objectclasses that are `groupOfUrls` (referred to as a "dynamic group") and convert it into a virtual static group or `groupOfUniqueNames` equivalent.

The DynamicGroups Workflow Element automatically processes any `memberURL` values and adds the results to the `uniqueMember` attribute. This dynamic object processing allows administrators to define groups that hold both static members and dynamic members while maintaining compatibility with applications that may not normally support the `groupOfUrls` objectclass.

For example, consider the search query as follows:

```
ldapsearch -D bindDN -q -b ou=groups,ou=airius,o=yourcompany.com -s sub "(memberurl=*)"

cn=test,ou=groups,ou=airius,o=yourcompany.com
cn=test
memberURL=ldap:///ou=accounting,o=yourcompany.com??sub?(&(objectclass=person)(obj
ectclass=organizationalperson))
objectclass=groupofUniqueNames
objectclass=groupOfUrls
objectclass=top
uniqueMember=cn=Paul Jacobs,ou=People,ou=Airius,o=yourcompany.com
uniqueMember=cn=Wendy Verbaas,ou=People,ou=Airius,o=YourCompany.com

cn=TestCheck,ou=groups,ou=airius,o=yourcompany.com
memberURL=ldap:///ou=alt bind,o=yourcompany.com??sub?(cn=*)
objectclass=groupofUniqueNames
objectclass=groupOfUrls
cn=TestCheck
```

In the above example, two groups are returned. The first group holds two static members and has a `memberURL` defining a particular directory subtree to also be members.

When the DynamicGroups Workflow is enabed, the same query results as follows:

```
./ldapsearch -D bindDN -q -b ou=groups,ou=airius,o=yourcompany.com -s sub "(cn=test)"
uniqueMember
cn=test,ou=groups,ou=airius,o=yourcompany.com
memberURL=ldap:///ou=accounting,o=yourcompany.com??sub?(&(objectclass=person)(obj
ectclass=organizationalperson))
objectclass=groupOfUniqueNames
objectclass=groupOfUrls
objectclass=top
cn=test
uniqueMember=cn=Paul Jacobs,ou=People,ou=Airius,o=yourcompany.com
uniqueMember=cn=Wendy Verbaas,ou=People,ou=Airius,o=YourCompany.com
uniqueMember=cn=Vipi Velasquez,ou=accounting,o=yourcompany.com
uniqueMember=cn=Preston Pena-Fernandez,ou=accounting,o=yourcompany.com
uniqueMember=cn=Andreas O'Hara,ou=accounting,o=yourcompany.com
uniqueMember=cn=Chitra Guenette,ou=accounting,o=yourcompany.com
...
uniqueMember=cn=Jim Ward,ou=accounting,o=yourcompany.com
```

The DynamicGroup Workflow Element expands the `memberURL` value (that is, it executes an LDAP search query with the base, scope, and filter specified in the `memberURL` and adds the returned DNs to the member attribute) if, and only if, the search filter specified in the client request does not return any entries from a backend directory.

To know more about how the DynamicGroup Workflow Element expands the `memberURL` , see Testing the DynamicGroups With and Without Expanding memberURL Attribute

12.5.8.2 Configuration Parameters for DynamicGroups Workflow Element

Learn about the configuration parameters for DynamicGroup Workflow Element and their functionality.

The following table describes the allowed properties and their description.

Table 12-6 Configuration Parameters for DynamicGroups Workflow Element

Property	Description	Mandatory or Optional	Default Value
check-url-base	Defines whether or not to honor the base and scope specified in `memberURL` while performing membership queries	Optional	False
enabled	Defines if the property is enabled or not for use in the server. If a Workflow Element is not enabled, then its contents are not accessible when processing operations.	Mandatory	True

Table 12-6 (Cont.) Configuration Parameters for DynamicGroups Workflow Element

Property	Description	Mandatory or Optional	Default Value
Expand-url	Defines whether the dynamic groups have to be expanded or not. If this value is not explicitly configured, it is implicitly calculated based on the requested attributes of a search request. For implicit calculation, if the requested attributes in a search query request is the `member/uniquemember` attribute, then the group is expanded. Otherwise, the group is not expanded. This explicitly configured value will override the implicit behavior.	Optional	-

Table 12-6 (Cont.) Configuration Parameters for DynamicGroups Workflow Element

Property	Description	Mandatory or Optional	Default Value
fetch-both-static-dynamic	Defines whether or not to fetch both static and dynamic groups which a user is a member of. If set to false, fetches only static groups to which a particular searched user belongs, and it does not fetch the dynamic group. If set to true, fetches both static and dynamic groups for the searched user.	Optional	False

Table 12-6 (Cont.) Configuration Parameters for DynamicGroups Workflow Element

Property	Description	Mandatory or Optional	Default Value
Global-search	Defines whether or not to perform global search while expanding dynamic group. If set to true, then the search would be performed against the entire network group which the current workflow element belongs to. If set to false, then the search would be performed starting from the next workflow element.	Optional	False

Table 12-6 (Cont.) Configuration Parameters for DynamicGroups Workflow Element

Property	Description	Mandatory or Optional	Default Value
member-attribute	Defines the name of the attribute that stores group member values. This attribute along with the appropriate objectclass would be added to the dynamic group entry after expanding memberURL attribute of the entry.	Optional	uniquemember
next-workflow-element	Defines the next workflow element in the chain of workflow elements	Mandatory	userRoot
user-search-base	Defines the optional value for a virtual search base that needs to override the base specified in an LDAP URL.	Optional	A valid DN

Note:

If `expand-url` is not defined, the search request for all attributes does not expand dynamic groups. For example, the dynamic group workflow element created with default options, the search query request for all attributes with filter containing `cn` does not expand dynamic groups.

See Configuring DynamicGroup Workflow Element to learn how to configure dynamic group workflow element.

12.6 Understanding the Global Index Catalog

A global index catalog can be used with a distribution deployment. If you are configuring a capacity based distribution, you must have a global index, with DN indexed.

The global index catalog maps the entries to the distribution partition in which the data is held. When the proxy receives a request from the client, the distribution looks up the attribute entry in the global index catalog, and forwards the client request to the correct partition. This diminishes the need for broadcasts. Moreover, if a modify DN request is made, the global index catalog will ensure that the entry is always found.

A global index catalog maps the entries based on specific attributes, such as employee number or telephone number. The value of the attribute to be indexed *must be unique* across all the entries. You cannot use a global index to map entries based on country, for example, as that information is not unique.

If you index an attribute whose values are not unique, the proxy server might be unable to return all the requested entries. Say, for example, that you index the `mail` attribute, whose values are not necessarily unique. You now add the following two entries in sequence:

- Entry 1, with `uid=user.1` and `mail=joe.smith@example.com` is sent to partition 1.
- Entry 2, with `uid=user.2` and `mail=joe.smith@example.com` is sent to partition 2.

In this situation, the global index `mail` keeps reference to the second entry only. A search with the filter `(mail=joe.smith@example.com)` will return only the second entry, `uid=user.2`.

A global index catalog can include several global indexes. Each global index maps a different attribute. For example, you can have one global index catalog called `GI-catalog`, which includes a global index mapping the entries based on the *telephone number* and one mapping the entries based on the *employee number*. This means that you can forward client requests to the right partition using either the telephone number or the employee number.

Global index catalogs and global indexes are created and configured using the `gicadm` command.

> **Note:**
>
> For more information see Configuring Global Indexes Using the Command Line and gicadm.

The global indexes can be populated with data from LDIF files. The data from one LDIF file can be split into partitions using the `split-ldif` command. For more information, see split-ldif.

A global index catalog should be replicated to avoid a single point of failure. For information on replicating the global index catalog, see Replicating Global Index Catalogs.

Consider the following example that uses a global index catalog for telephone numbers.

A typical example of a unique attribute which can be used to create a global index is a telephone number: the value of the attribute is unique, that is, only one person (employee, for example) can have that telephone number.

In the example below, the entries in the database have been split based on the telephone number. The global index includes the following information:

Value	Partition ID
4011233	1
4011234	1
7054477	2

The global index does not store the name of the employees, location, and other attribute values that may be associated to the telephone number. It only maps the attribute indexed to the partition. The data associated to the indexed value (here telephone number) is stored in the remote LDAP server.

If an employee has multiple phone numbers, these are regarded as multi-valued entries. In this case, if the global index is created based on the telephone number, there will be two global index entries that will result in finding one employee, say Ben Brown.

In the example above, employee Ben Brown could have both telephone number `4011233` and `7054477` assigned to him. In this case, a search on one of Ben Brown's telephone number would return the correct partition, and all the information associated to the telephone number, including the name Ben Brown, regardless that he has two phone numbers attributed to him.

12.7 Understanding the Transformation Framework

Oracle Unified Directory supports transformation of data through the definition of workflow elements. By creating an instance of workflow element you can display physical data in a different way.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

This section describes how transformation in Oracle Unified Directory occurs and contains the following topics:

- Overview of Transformation
- Components of Transformation
- Examples of Transformation Use Case Configuration

> **Note:**
>
> For information about configuring transformation, see Configuring Transformations.

12.7.1 Overview of Transformation

A transformation performs a specific action in a certain direction. You must specify the transformations that you need and define these on an existing workflow element.

The data structure of an LDAP client application may differ from the data structure of the LDAP repository. They may differ on the schema (different types of attribute in the entries) or the values (same attribute name with different semantic of values). This is where you need transformation.

The topics in this section include:

- Overview of Transformation Models
- Implementing Transformation in Oracle Unified Directory

12.7.1.1 Overview of Transformation Models

The direction of transformation (that is whether the transformation is applied during the request, during the response, or both) determines the transformation model.

Transformations can be categorized into the following types:

- **Read transformations** (outbound transformations): For more information, see Understanding Read Transformations.
- **Write transformations** (inbound transformations): For more information, see Understanding Write Transformations.
- **Mapping transformations** (bidirectional transformations): For more information, See Understanding Mapping Transformations.

12.7.1.1.1 Understanding Read Transformations

The read transformation is the most common transformation. A read transformation is applied only during the response to a request. No transformation is applied during the request and the physical data is not changed.

Figure 12-22 illustrates the concept of a read transformation.

Figure 12-22 Read Transformation

Consider a scenario of an organization that has a legacy application whose function is to display person entries. The application does not support entries that do not contain an `email` attribute. The physical data source has been upgraded and the `email` attribute no longer exists for person entries.

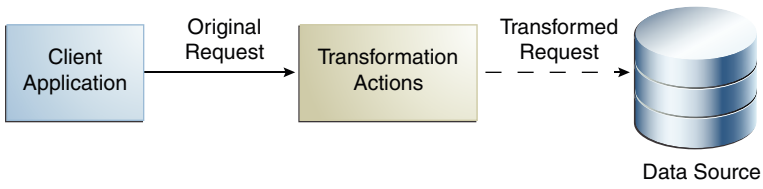
You must apply a transformation here, which is to add the `email` attribute during the search response. This transformation changes the entry that is read from the data source and adds an `email` attribute whose value is `firstname.surname@mycompany.com`. No reverse transformation is required and the physical data is not changed.

12.7.1.1.2 Understanding Write Transformations

A write transformation is applied during the request, but not during the response. A write transformation modifies data provided by the client before storing it in the back end.

Figure 12-23 illustrates the concept of a write transformation.

Figure 12-23 Write Transformation



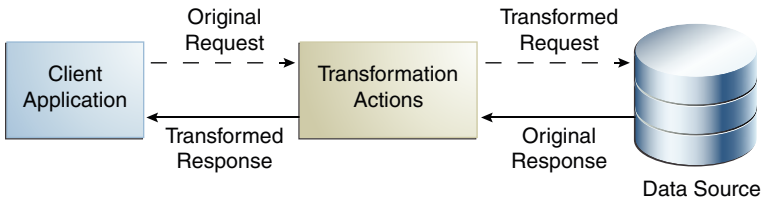
Consider a scenario of an organization that has a legacy application whose function is to add person entries to a data source. The application adds the entries without the `email` attribute. The physical data source has been upgraded and the `email` is now a mandatory attribute for person entries. You must apply a transformation here, which is to add the `email` attribute during the add request. This transformation changes the entry that is written to the database. No reverse transformation is required.

12.7.1.1.3 Understanding Mapping Transformations

The mapping transformation is the most common transformation. It is bidirectional in the sense that it is first applied during the request, and the reverse is applied during the response. These transformations are called mappings, because an attribute or entry in the physical data view maps to an attribute or entry in the virtual data view. Mapping transformations enable you to process existing values before assigning them to a DN component, an attribute type or value, or an object class.

Figure 12-24 illustrates the concept of a mapping transformation.

Figure 12-24 Mapping Transformation



Consider a scenario of an organization, which has a physical data source that contains entries with the attributes `surname` and `firstname`. The organization has a client application that requires entries to have a `cn` (common name) attribute of the form `firstname surname`.

The client application sends a search request for an entry of the form `cn=Joe Smith`. A transformation is defined that extracts the firstname and surname during this request and

transforms the request to one of the form `surname=Smith, firstname=Joe`. The corresponding entry is located in the data source. Before returning this entry to the client application, the inverse transformation is performed. The client application receives the entry as `cn=Joe Smith`, which it understands.

This request is transformed to be of the form `surname=Smith, firstname=Joe`.

12.7.1.2 Implementing Transformation in Oracle Unified Directory

Oracle Unified Directory is an LDAP server that supports transformations in a proxy server.

To implement transformations, you must:

- Create an instance of a workflow element of type **transformations**.
- Insert the transformation workflow element in the desired workflow elements list.

> **Note:**
>
> - For more information about the architecture of Oracle Unified Directory, see Overview of Oracle Unified Directory Architecture.
> - For more information about configuring transformations, see Configuring Transformations.

A transformation workflow element instance is essentially a data view on which certain transformation actions are defined.

12.7.2 Components of Transformation

Understand about the components for configuring the workflow elements for transformation.

The topics in this section are:

- Overview of Transformation Types
- Overview of Transformation Conditions
- Defining Attribute Values for Transformation

12.7.2.1 Overview of Transformation Types

To configure the workflow element of transformation types use the following set of transformations:

- `addOutboundAttribute` Transformation Type
- `filterOutboundAttribute` Transformation Type
- `addInboundAttribute` Transformation Type
- `filterInboundAttribute` Transformation Type
- `mapAttribute` Transformation Type
- map-object-class Transformation Type
- tokenize-attribute Transformation Type

> **Note:**
>
> Here:
>
> - **Client side:** Refers to the side where the Oracle Unified Directory server interacts with the client application.
> - **Source side:** Refers to the side where the Oracle Unified Directory server interacts as a data server with its local data source, or as a proxy server with a remote server.
> - **Inbound direction:** Refers to the direction where transformations are applied from the client to the source.
> - **Outbound direction:** Refers to the direction where transformations are applied from the source to the client.

12.7.2.1.1 `addOutboundAttribute` Transformation Type

The `addOutboundAttribute` transformation adds a virtual attribute or value(s) to entries returned to the client during a SEARCH operation, when the list of attributes in the request is either undefined (all) or when it contains this attribute.

When you cannot determine if an entry already contains a virtual attribute, the conflict-behavior parameter decides which of the following policy will apply:

- The virtual value is not added
- The virtual value is added and merged with the existing values
- The virtual value replaces the existing one

If you are aware that the virtual attribute is searchable in the source repository, which implies some entries in the source repository contain the virtual attribute and searches are optimized on this attribute, and if the flag `virtual-in-source` is set then the transformation process forwards the virtual attribute to the source repository in the `SEARCH REQUEST` filter. Usually, the virtual attribute is not forwarded to the source repository. When it is set to `FALSE`, search requests are optimized for common cases, which implies virtual attributes not expected to be in the source repository.

> **Note:**
>
> You must keep in mind that the source schema check is applied when the virtual attribute is expected to appear in ADD or MODIFY requests. Therefore, it is recommended to configure the schema of the source to accept the virtual attribute. Otherwise, disable schema checking.

Table 12-7 describes the parameters of `addOutboundAttribute` transformation type.

Table 12-7 Parameters of addOutboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Name of the client virtual attribute and the value definitions of the client virtual attribute	`client-attribute`	**S**	**M**	string For more information, see Defining Attribute Values for Transformation. For example, `displayName=%cn%` publishes the attribute `displayName` with value of `cn`.

Conflict behavior policy	`conflict-behavior`	**S**	**O** [default=`merge-real-and-virtual`]	`merge-real-and-virtual`
				`real-overrides-virtual`
				`virtual-overrides-real`

Virtual in source policy	`virtual-in-source`	**S**	**O** [default = `FALSE`]	`TRUE, FALSE`
Condition based on a filter that the entry must match	`entry-match-filter`	**S**	**O** [default = apply to all entries processed by the workflow element]	LDAP filter
Condition based on DN that must be an ascendant	`entry-parent-suffix`	**M**	**O** [default = apply to all requests processed by the workflow element]	DN
Condition to exclude operations in the operation processing	`excluded-operation`	**M**	**O** [default = apply to all LDAP operations]	enumerated (ADD, MODIFY...)

12.7.2.1.2 `filterOutboundAttribute` Transformation Type

The `filterOutboundAttribute` transformation removes an attribute or value(s) from entries received from the source before sending to the client.

Table 12-8 describes the parameters of `filterOutboundAttribute` transformation type.

Table 12-8 Parameters of FilterOutboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Name of the source attribute and the value definitions of the source attribute	`source-attribute`	**S**	**M**	string For more information, see Defining Attribute Values for Transformation. For example, `certificate=verisign` filters the `verisign` value from the `certificate` attribute.
Condition based on a filter that the entry must match	`entry-match-filter`	**S**	**O** [default = apply to all entries processed by the workflow element]	LDAP filter

Table 12-8 (Cont.) Parameters of FilterOutboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Condition based on DN that must be an ascendant	`entry-parent-suffix`	**M**	**O** [default = apply to all requests processed by the workflow element]	DN
Condition to exclude operations in the operation processing	`excluded-operation`	**M**	**O** [default = apply to all LDAP operations]	enumerated (ADD, MODIFY...)

12.7.2.1.3 `addInboundAttribute` Transformation Type

The `addInboundAttribute` transformation adds a virtual attribute or value(s) to entries received from the client while performing the ADD operation before forwarding the data to the source.

When you cannot determine if an entry already contains a virtual attribute, the conflict-behavior parameter decides which of the following policy will apply:

- The virtual value is not added
- The virtual value is added and merged with the existing values
- The virtual value replaces the existing one

Table 12-9 describes the parameters of `addInboundAttribute` transformation type.

Table 12-9 Parameters of addInboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Name of the source virtual attribute and the value definitions of the source virtual attribute	`source-attribute`	**S**	**M**	string For more information, see Defining Attribute Values for Transformation. For example, `email={%cn%.%sn%@mycompany.com}` writes the attribute `email` with value derived from attributes `cn` and `sn`.

Conflict behavior policy	`conflict-behavior`	**S**	**O** [default=`merge-real-and-virtual`]	`merge-real-and-virtual`
				`real-overrides-virtual`
				`virtual-overrides-real`

Condition based on a filter that the entry must match	`entry-match-filter`	**S**	**O** [default = apply to all entries processed by the worflow element]	LDAP filter
Condition based on DN that must be an ascendant	`entry-parent-suffix`	**M**	**O** [default = apply to all requests processed by the workflow element]	DN

Table 12-9 (Cont.) Parameters of addInboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Condition to exclude operations in the operation processing	`excluded-operation`	**M**	**O** [default = apply to all LDAP operations]	enumerated (ADD, MODIFY)

12.7.2.1.4 `filterInboundAttribute` Transformation Type

The `filterInboundAttribute` transformation removes an attribute or value(s) from entries (and modifications) received from the client on a ADD (and MODIFY) before forwarding to the source.

Table 12-10 describes the parameters of `filterInboundAttribute` transformation type.

Table 12-10 Parameters of FilterInboundAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Name of the client virtual attribute and the value definitions of the client virtual attribute	`client-attribute`	**S**	**M**	string For more information, see Defining Attribute Values for Transformation. For example, `certificate=verisign` filters the value `verisign` of the attribute `certificate`. Similarly, `secondarylocation=%primarylocation%` filters the values of `secondarylocation` when it matches the values of `primarylocation`.
Condition based on a filter that the entry must match	`entry-match-filter`	**S**	**O** [default = apply to all entries processed by the worflow element]	LDAP filter
Condition based on DN that must be an ascendant	`entry-parent-suffix`	**M**	**O** [default = apply to all requests processed by the workflow element]	DN
Condition to exclude operations in the operation processing	`excluded-operation`	**M**	**O** [default = apply to all LDAP operations]	enumerated (ADD, MODIFY...)

12.7.2.1.5 `mapAttribute` Transformation Type

The `mapAttribute` transformation can rename or revalue a client attribute to one source attribute in both directions.

Table 12-11 describes the parameters of `mapAttribute` transformation type.

Table 12-11 Parameters of mapAttribute Transformation Type

Parameter	dsconfig CLI	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
Name of the client attribute and the value definitions of the mapping from the client virtual attribute to the source virtual attribute	`client-attribute`	**S**	**M**	string For more information, see Defining Attribute Values for Transformation. For example, `displayName=%cn%` publishes `displayName` attribute replacing it with the value of `cn` attribute, and writes `cn` attribute replacing it with the value of `displayName` attribute.
Virtual in source policy	`virtual-in-source`	**S**	**O** [default = `FALSE`]	`TRUE, FALSE`

Conflict behavior policy	`conflict-behavior`	**S**	**O** [default=`merge-real-and-virtual`]	`merge-real-and-virtual`
				`real-overrides-virtual`
				`virtual-overrides-real`

Condition based on a filter that the entry must match	`entry-match-filter`	**S**	**O** [default = apply to all entries processed by the worflow element]	LDAP filter
Condition based on DN that must be an ascendant	`entry-parent-suffix`	**M**	**O** [default = apply to all requests processed by the workflow element]	DN
Condition to exclude operations in the operation processing	`excluded-operation`	**M**	**O** [default = apply to all LDAP operations]	enumerated (ADD, MODIFY,...)

12.7.2.1.6 Map Object Class Transformation Type

The Map Object Class transformation creates mapping that can make one objectClass appear like another objectClass. For example a source objectClass `inetOrgPerson` can appear like a client objectClass `user`. This ability is useful when an application expects a particular objectClass/attributes, but the directory does not support that.

The Map Object Class transformation can perform different types of manipulations based on the selected configuration parameters. These manipulations are attribute mapping, objectclass mapping, adding or removing attributes conditional on objectclass, and filtering object classes during write or modify operations. The Map Object Class transformation takes multiple parameters. The `map-objectclass` is a mandatory parameter in this transformation. The other parameters are optional.

Table 12-12 Configuration Parameters for Map Object Class transformation

Parameter	Description	Multi (M) / Single (S) Valued	Mandatory (M) / Optional (O)	Value
`map-objectclass`	It maps a client objectclass to a source objectclass during add, modify (only when `filter-objectclass-on-modify` value is false) and search operations. Its syntax is `client-objectclass=source-objectclass`. Example: `user=inetOrgPerson` The client objectclass `user` is converted to source objectclass`inetOrgPerson` when sent to the source repository. And it's converted back from `inetOrgPerson` to `user` when read from the source repository and sent to the client application.	M	M	objectclass=objectclass

`add-attribute`	It adds an attribute to the entry during the add operation. You may specify a conditional source objectclass name to add an attribute to that entry. The attribute will be added only if the entry contains the source objectclass. In that case, you must prefix the objectclass name to the attribute name with '`:`' as the separator.
	Example:
	`inetOrgPerson:userAccountControl=546`
	It will add `userAccountControl` attribute to the entry if the entry contains the source objectclass `inetOrgPerson`.
	`inetOrgPerson:displayName=%cn%`
	It will add `displayName` attribute, with value same as `cn` attribute, to the entry if the entry contains the source objectclass `inetOrgPerson`.
		M	O	objectclass:attribute

Table 12-12 (Cont.) Configuration Parameters for Map Object Class transformation

Parameter	Description	Multi (M) / Single (S) Valued	Mandatory (M) / Optional (O)	Value
`entry-match-filter`	Specifies the filter that the entry must match for this transformation to apply.	S	O	String
`entry-parent-suffix`	Specifies an ascendant of the operation DN for this transformation to apply.	M	O	DN
`excluded-operation`	Specifies operations on which this transformation never apply. This allows to restrict the regular behavior of this transformation.	M	O	Operation

`filter-attribute`	It will remove an attribute from the entry during add, modify, and search operations. You may specify a conditional source objectclass name to remove the attribute from the entry. The attribute will be removed only if the entry contains the source objectclass. In that case, you must prefix the source objectclass name to the attribute name with '`:`' as the separator.
	Example:
	`accountname`
	It will remove the `accountname` attribute from the entry during add, modify, and search operations.
	`inetOrgPerson:memberof`
	It will remove the `memberof` attribute from the entry during add and modify operations. The attribute will be removed only if the entry contains the source objectclass `inetorgperson`. However, it will not remove the `memberof` attribute from the entry while reading it from the source.
		M	O	objectclass:attribute

`filter-objectclass-on-modify`	It's a flag to determine whether or not to remove object classes during modify operations. Supported values include `true` (remove changes) and `false` (do not remove changes), where `true` is the default value.	S	O	Boolean

Table 12-12 (Cont.) Configuration Parameters for Map Object Class transformation

Parameter	Description	Multi (M) / Single (S) Valued	Mandatory (M) / Optional (O)	Value
`filter-objectclass-on-write`	It remove an objectclass during add and modify operations. For instance, Microsoft Active Directory for Windows 2000 does not allow auxiliary object classes to be listed while adding an entry, while Microsoft Active Directory and ADAM for Windows Server 2003 does allow for auxiliary classes to be listed. For example, if the value is specified as `person`, then the `person` objectclass will be removed from the entry during add and modify operations.	M	O	Objectclass

Table 12-12 (Cont.) Configuration Parameters for Map Object Class transformation

Parameter	Description	Multi (M) / Single (S) Valued	Mandatory (M) / Optional (O)	Value
`map-attribute`	It maps a client attribute to a source attribute during add, modify and search operations. It's syntax is `client-attribute=source-attribute`. Example: `displayName=%givenName%` The value of the client attribute `displayName` is assigned to source attribute `givenName` when sent to the source repository. It's reverse assigned from the source attribute `givenName` to client attribute `displayName` when its read from the source repository. This parameter also supports value mapping syntax: `o=%ou%(US,SFO)(IN,BGL)`. Here, the client attribute `o` is mapped to the source attribute `ou`. If the value of `o` is `US`, then `ou` is assigned `SFO`. Similarly, if the value of `o` is `IN`, then `ou` is assigned `BGL`. It's reverse assigned from the source attribute `ou` to client attribute `o` when read from the source repository.	M	O	Attribute

12.7.2.1.7 `tokenize-attribute` Transformation Type

The `tokenize-attribute` transformation allows you to apply the transformation on the attribute which has values separated with delimiter.

For example, an attribute `attr:a,b,c` is split it into multiple values by delimiter and when a `tokenize-attribute` transformation is applied, it returns as a true multi-valued attribute. Such as:

```
attr:a
attr:b
attr:c
```

Table 12-13 describes the parameters of `tokenize-attribute` Transformation Type.

Table 12-13 Parameters of tokenize-attribute Transformation Type

Parameter	Description	Multi (M) / Single (S) Valued	Optional (O) / Mandatory (M)	Values
`multivalued-attribute`	Specifies the name of the attribute whose value will be tokenized with the delimiter as multiple values.	M	M	Attribute
`tranformation-value-delemiter`	Specifies the delimiter to tokenize the attribute value to multiple values.By default the value will be ","	S	O	String

12.7.2.2 Overview of Transformation Conditions

You can configure the **Transformations** workflow element with a set of conditions. Conditions are properties (attributes) that can be set either on a `transformations-workflow-element` or on an individual `transformation`.Transformation works only when LDAP request matches all conditions and all conditions set at the level of workflow element.

The following conditions are applicable for implementing transformation:

- You can configure conditions to rules whether transformations apply or not.
- You can set conditions on the transformations-workflow-element. In this situation, conditions apply for all transformations set on the workflow-element and they are evaluated prior to eventually processing each transformation.
- You can set conditions on each individual transformation and they are evaluated prior to eventually processing this transformation.

In this sense, conditions can be broadly categorized as follows:

- About Parent Suffix
- About Entry Match Filter
- About Excluded LDAP Operation

12.7.2.2.1 About Parent Suffix

This condition is applicable for transformations applied only for LDAP operations that target an entry for which name is under one of the parent suffixes specified.

When no condition of this type is configured, then transformation applies to all entries processed.

12.7.2.2.2 About Entry Match Filter

This condition is applicable for transformations applied on LDAP operations only for entries that match the provided filter.

When no condition of this type is configured, then transformation applies to all entries processed.

12.7.2.2.3 About Excluded LDAP Operation

This condition specifies a list of multi-valued attributes, where each attribute is an LDAP operation that should *not* be impacted by the transformation. It allows you to disable the action of the transformation (when it has one) on each LDAP protocol message.

When no condition of this type is configured, then transformation applies to all LDAP operations normally impacted by this type of transformation.

12.7.2.3 Defining Attribute Values for Transformation

An attribute value allows you to define the value of a virtual attribute during transformation. This value can either be a default value, or rule that creates the value from other attribute values.

For `addInboundAttribute`, `addOutboundAttribute`, and `mapAttribute`, you must configure the values of the virtual attribute added. For `filterInboundAttribute` and `filterOutboundAttribute`, the values you intend to filter *may* be configured.

An attribute can derive its value from the following:

- Using Constant
- Using Value of Another Attribute
- Using Regular Expressions
- Using Values Mapping
- Using Multi-valued Virtual Attributes
- Using Tokenized Attribute Transformation

12.7.2.3.1 Using Constant

It is used to generate an attribute with a static default value or to filter a static value of an attribute.

For example, the property `source-attribute:mycompany=Acme` is used to provide a default company name.

```
dsconfig create-transformation \
--type add-inbound-attribute \
--set source-attribute:mycompany=Acme \
--transformation-name virtDeptName \
```

12.7.2.3.2 Using Value of Another Attribute

It is used to create a new attribute from an existing attribute in the entry that is being processed or to filter a value taken from another attribute using the `%inputAttrName%` syntax.

For example, the property `source-attribute:displayName=%cn%` specifies that the value of the new attribute must be taken from the value of the `cn` attribute.

```
dsconfig create-transformation \
--type add-inbound-attribute \
--set source-attribute:displayName=%cn% \
--transformation-name virtDeptName \
```

> **Note:**
>
> You must keep in mind that another virtual attribute generated in the same `transformations-workflow-element` should not be referenced, because the evaluation order is not guaranteed.

12.7.2.3.3 Using Regular Expressions

It is used to create an attribute value or to filter an attribute value by manipulating the value of an existing attribute using the `{expression}` syntax.

For example, the property `client-attribute:mail={%cn%.%sn%@mycompany.com}` is a regular expression that is used for deriving an attribute by combining the values of existing attributes.

```
dsconfig create-transformation \
--type add-outbound-attribute \
--set client-attribute:mail={%cn%.%sn%@mycompany.com} \
--transformation-name virtDeptName \
```

12.7.2.3.4 Using Values Mapping

It is used for defining virtual values as a mapping of values of another attribute using the `virtAttrName=%refAttrName%(virtValue1,refValue1)(virtValue2,refValue2)` syntax.

For the `virtAttrName` parameter, the transformation adds or filters values extracted from `refAttrName`. If `refAttrName` matches `refValue1`, then transformation processes either add or filter for `virtValue1`. In the values provided, characters `'(', ')', ','` and `'\'`must be escaped using `'\'` character.

For example, consider an organization with several departments where department name is returned for the retrieved department ID, such as Department:1–Marketing, 2–Sales, 3–Finance and so on. But, when `deptId` is `1`, the value returned for `deptName` is `Marketing`. When `deptId` is `2`, the value for `deptName` is `Sales`. Similarly, when `deptId` is `3`, the value returned for `deptName` is `Finance`.

```
dsconfig create-transformation \
--type add-outbound-attribute \
--set client-attribute:deptName=%deptId%(Marketing,1)(Sales,2)(Finance,3) \
--transformation-name virtDeptName
```

12.7.2.3.5 Using Multi-valued Virtual Attributes

It is used to specify a virtual multi-valued attribute using the `virtAttrName=virtAttrValue1=virtAttrValue2=` syntax.

```
dsconfig create-transformation \
--type add-outbound-attribute \
--set client-attribute:countriesResp=France=Germany=Italy \
--transformation-name virtCountriesRep
```

12.7.2.3.6 Using Tokenized Attribute Transformation

Tokenization is the process of taking the input text (such as a sentence) and then splitting it into individual terms, called tokens. Tokenize attribute transformation applies the

transformation on the attribute which has values separated with delimiter then split it into the multiple values by delimiter and returns as a true multi-valued attribute.

In the values provided, delimiter is configurable. By default the delimiter is ‘,’. If the delimiter character itself is part of the value, then it must be escaped using ‘\’ character.

For example, consider the following multi valued attribute for `telephoneNumber`.

Note that the `user.0` has the following details stored:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectClass: inetorgperson
uid: user.0
telephoneNumber: +1 685 622 6202, +1 785 788 4567, +1 456 765 3456
```

Apply tokenize-attribute transformation on `telephoneNumber` attribute:

```
./dsconfig create-transformation \
--type tokenize-attribute \
--set multivalued-attribute:telephoneNumber \
--transformation-name tokentrans \
```

Perform a search on the attribute `telephoneNumber`:

```
./ldapsearch -h hostname -p 20389 -D "cn=directory manager" -w password -
s sub -b "dc=example,dc=com" "(uid=user.0)" telephoneNumber uid
```

Here, the tokenize-attribute transformation applies on the `telephoneNumber` attribute and tokenizes the value with delimiter and gives the value as true multi-valued attribute.It returns `user.0` which has multiple values for `telephoneNumber`:

```
dn: uid=user.0,ou=People,dc=example,dc=com
telephoneNumber: +1 685 622 6202
telephoneNumber: +1 785 788 4567
telephoneNumber: +1 456 765 3456
uid: user.0
```

12.7.3 Examples of Transformation Use Case Configuration

Understand how to map activation or deactivation for a specific back end directory and how to map object classes.

The following topics illustrate practical uses for transformation, and provide example configurations:

- Mapping Activation or Deactivation for a Specific Back End Directory
- Mapping Object Classes by Using map-attribute Transformation Type
- Mapping Object Classes by Using map-object-class Transformation Type
- Adding Attributes to Source Object Class by Using map-object-class Transformation Type
- Filtering Attributes from Source Object Class by Using map-object-class Transformation Type

12.7.3.1 Mapping Activation or Deactivation for a Specific Back End Directory

This configuration is useful when an application has its own user activation attribute and values, which is different from the back end user activation attribute and values, and a mapping for read and write operations is required.

In the following example, the attribute `myuseraccountcontrol` with values `activated` and `deactivated` transforms to the back-end attribute `nsAccountLock` with values `false` and `true` for a DSEE (SunONE) back end.

```
$ ./dsconfig -X -n -Q -p 1444 -D cn=directory manager -j pwdfile
 create-transformation --transformation-name mapactivate --type map-attribute
--set client-attribute:myuseraccountcontrol="%nsAccountLock%(activated,false)
(deactivated,true)"
```

In the following example, the attribute `myuseraccountcontrol` with values `activated` and `inactivated` transforms to back-end attribute `userAccountControl` with values `544` and `546` for an Active Directory back end.

```
$ ./dsconfig -X -n -Q -p 1444 -D cn=directory\ manager -j pwdfile
 create-transformation --transformation-name mapactivate --type map-attribute
 --set client-attribute:myuseraccountcontrol="%userAccountControl%
(activated,544)(deactivated,546)"
```

12.7.3.2 Mapping Object Classes by Using map-attribute Transformation Type

This configuration is useful when an application has an objectclass with same meaning as an objectclass on the back-end server, but the two objectclasses have different names. A mapping on the objectclass name is required on read and write operations.

In the following example, a search comes in from the client with filter `objectClass=User`, and you want to transform that filter to `objectClass=inetOrgUser`. When an entry is returned to the client, if the entry is stored with the `objectClass=inetOrgUser`, then the entry is mapped to objectClass `User`.

```
$ ./dsconfig -X -n -Q -p 1444 -D cn=directory manager -j pwdfile
 create-transformation --transformation-name mapoc --type map-attribute
--set client-attribute:objectClass="%objectClass%(User,inetOrgUser)"

$ ./dsconfig -X -n -Q -p 1444 -D cn=directory manager -j pwdfile
 create-workflow-element --type transformations  --element-name trsfwfe
--set enabled:true --set next-workflow-element:userRoot --set transformation:mapoc
$ ./dsconfig -X -n -Q -p 1444 -D cn=directory manager -j pwdfile set-workflow-prop
 --workflow-name userRoot0 --set workflow-element:trsfwfe
```

> **Note:**
>
> It is recommended to use `map-object-class` Transformation Type to map object classes.

12.7.3.3 Mapping Object Classes by Using map-object-class Transformation Type

The Map Object Class transformation creates mapping that can make one objectClass appear like another objectClass. For example a source objectClass `inetOrgPerson` can appear like a client objectClass `user`. You can create a new Map Object Class transformation by using the

dsconfig command. The `map-objectclass` parameter is a mandatory parameter to create a Map Object Class transformation.

To create the new transformation:

- Run the `dsconfig` command as shown below:

```
$ ./dsconfig create-transformation --type map-object-class --
transformation-name ocm2 --set map-objectclass:user=inetOrgPerson -h
localhost -p 10444 -D "cn=Directory Manager" -j /pwd-file1.txt


>>>> Configure the properties of the Map Object Class Transformation

        Property                      Value(s)
        ----------------------------------------------
    1)  add-attribute                 -
    2)  entry-match-filter            -
    3)  entry-parent-suffix           -
    4)  excluded-operation            -
    5)  filter-attribute              -
    6)  filter-objectclass-on-modify  true
    7)  filter-objectclass-on-write   -
    8)  map-attribute                 -
    9)  map-objectclass               user=inetOrgPerson

    ?)  help
    f)  finish - create the new Map Object Class Transformation
    q)  quit

Enter choice [f]: f

The Map Object Class Transformation was created successfully
```

> **Note:**
>
> It is recommended to use `map-object-class` Transformation Type to map object classes.

12.7.3.4 Adding Attributes to Source Object Class by Using map-object-class Transformation Type

This example will add `displayName` attribute, with value same as `cn` attribute, to the entry if the source objectclass is `inetOrgPerson`.

```
dsconfig set-transformation-prop --transformation-name ocm2 --set add-
attribute:inetOrgPerson:displayName=%cn% -n -X -h localhost -p 10444 -D "cn=Directory
Manager" -j /pwd-file1.txt
```

12.7.3.5 Filtering Attributes from Source Object Class by Using map-object-class Transformation Type

This example will remove the `memberof` attribute from the entry with source objectclass `inetOrgPerson`, during add and modify operations. However, it will not remove the `memberof` attribute from the entry while reading it from the source.

```
dsconfig set-transformation-prop --transformation-name ocm2 --add filter-
attribute:inetOrgPerson:memberof -n -X -h localhost -p 10444 -D "cn=Directory Manager" -
j /pwd-file1.txt
```

13

Understanding Identity Mapping in Oracle Unified Directory

Learn about Identity Mapping in Oracle Unified Directory, supported Identity Mappers, its components and priorities and how to configure Identity mappers from the following sections. This section has the following topics that give a description of identity mapping in Oracle unified directory:

- Overview of Identity Mappers
- Supported Identity Mappers
- Components of Identity Mappers
- Configuring Identity Mappers
- Understanding the Difference between Generic and GSSAPI Identity Mappers
- Defining Priority in Identity Mappers

13.1 Overview of Identity Mappers

Identity Mappers are responsible for establishing a mapping between an identifier string provided by a client, and the entry for the user that corresponds to that identifier.

Identity Mappers are used to process several SASL mechanisms to map an authentication ID (for instance, a Kerberos principal when using GSSAPI) to a directory user. They are also used when processing requests with the proxied authorization control.

Oracle Unified Directory supports multiple SASL identity mappers. For example, you can define Identity Mapper1 for a user `xyz` and Identity Mapper2 for the remaining users. This is beneficial when using GSSAPI where users with different domains, such as `@example.com` and `@oracle.com` require different identity mappers.

Oracle Unified Directory also provides support for an identifier string that is a bind ID and not a DN. However, this is applicable for simple binds only. The key idea is that a client should be able to specify any attribute in the simple bind that is allowed by the corresponding Identity Mapper. Consider the following examples:

```
ldapsearch -D "user@example.com" -w password -b "" objectclass=*
```

In this example, bind ID is the e-mail ID of the user.

13.2 Supported Identity Mappers

Identity Mappers are responsible for establishing a mapping between an identifier string provided by a client, and the entry for the user that corresponds to that identifier.

The following topics describe the Identity Mappers that are available in the server:

- About the Exact Match Identity Mapper
- About the Match And Replace Identity Mapper

13.2.1 About the Exact Match Identity Mapper

The Exact Match Identity Mapper maps an identifier string to a user entry by searching for the entry containing a specified attribute whose value is the provided identifier.

For example, the user name provided by the client for DIGEST-MD5 authentication must match the value of the `uid` attribute.

> **Note:**
>
> You must specify this attribute in the identity mapper configuration.

This mapper is primarily used in simple binds and all SASL binds except GSSAPI.

13.2.2 About the Match And Replace Identity Mapper

The Match And Replace Identity Mapper provides a way to use a regular expression to translate the provided identifier when searching for the appropriate user entry.

For example, you can use this mapper if you expect the provided identifier to be an e-mail address or Kerberos principal, but only the user name (the part preceding the `@` symbol) should be used in the mapping process.

> **Note:**
>
> A replacement is made only if all or part of the provided ID string matches the given match pattern. If no part of the ID string matches the provided pattern, the given ID string is used without any alteration.

This mapper is primarily used in GSSAPI binds.

13.3 Components of Identity Mappers

Certain components have a direct aggregation relation to Identity Mappers.

The following topics describe the components that have a direct aggregation relation to Identity Mappers:

- About the Role of Global Configuration in Identity Mappers
- About the Role of Network Group in Identity Mappers

13.3.1 About the Role of Global Configuration in Identity Mappers

The Global Configuration contains properties that affect the overall operation of the Oracle Unified Directory.

13.3.2 About the Role of Network Group in Identity Mappers

The Network Group is used to classify incoming client connections and route requests to workflow.

13.4 Configuring Identity Mappers

You must configure Identity Mappers for Network Group and Global Configuration instances.

Each Network Group has one or more Identity and Certificate mappers, which are used to map identities specific to that network group. If an identity or certificate mapper is not defined at the network-group level, then a global identity mapper is used as the default setting.

This section contains the following topics:

- Configuring Global Identity Mappers
- Configuring Network Group Identity Mappers

13.4.1 Configuring Global Identity Mappers

Identity mappers are configured by default at the global level.

Follow the command below to configure an identity mapper globally:

```
dsconfig set-global-configuration-prop --add "generic-identity-mapper:Exact Match"
```

The preceding command is based on the assumption that the `Exact Match` identity mapper already exists. This identity mapper is provided by default in the configuration.

13.4.2 Configuring Network Group Identity Mappers

You can configure the generic-identity-mapper for an existing default network group called `network-group`.

Follow the command below to configure the generic-identity-mapper for `network-group`:

```
dsconfig set-network-group-prop --group-name network-group --set "generic-identity-
mapper:Exact Match"
```

The preceding command is based on the assumption that the `Exact Match` identity mapper already exists. This identity mapper is provided by default in the configuration.

13.5 Understanding the Difference between Generic and GSSAPI Identity Mappers

Normally, one identity mapper is defined per network group. The generic-identity-mapper defines an identity mapper that applies to all but GSSAPI binds. The gssapi-identity-mapper defines the one that applies to GSSAPI binds only.

As described earlier, the exact match and replace identity mappers are generally used as generic-identity-mapper and gssapi-identity-mapper respectively. However, you can select a different combination based on your requirement.

13.6 Defining Priority in Identity Mappers

An identity mapper is selected based on the regex pattern. Therefore, there is a possibility that a conflict might arise when multiple identity mappers are defined. So, it becomes imperative to define the order in which identity mappers are evaluated in the network group.

You can define priorities for the conflicting identity mappers to resolve this conflict. If a conflict arises, the identity mapper with the lowest priority is selected and used for mapping. If identity mappers have equal priority, then the behavior is undefined.

Run the following command to define priority:

```
dsconfig -h hostname -p admin_port -D USER set-identity-mapper-prop --mapper-name "Exact Match" --set "priority:2"
```

A lower priority value implies higher priority. Priority for network groups is also determined in a similar fashion.

14

Understanding Data Encryption in Oracle Unified Directory

Understand about data encryption in Oracle Unified Directory from the following topics. Get an insight into basic encryption concepts, supported features, and basic configuration tasks.

14.1 What is Attribute Encryption?

Encryption is a mechanism that converts plaintext data into something unreadable, called *ciphertext*, to prevent unauthorized access to sensitive data. Decryption is the process in which the ciphertext is converted back to plaintext.

Oracle Unified Directory is a next-generation unified directory solution that integrates storage, synchronization, and proxy functionality to help you manage the critical identity information that drives your business applications. This data might contain sensitive information that should be available only to the intended recipient. Oracle Unified Directory offers mechanisms; such as access control rules, password authentication, and SSL to secure access to your data. Your data might also contain some extremely sensitive information, such as credit card numbers and SSN numbers. For this type of data, standard measures alone are not sufficient to prevent unauthorized access because the information is stored as human readable plaintext within the database. If an invader gains access to your server storage files and uses this information to their advantage, then the loss could present a high security risk.

Oracle Unified Directory provides an attribute encryption feature that enables you to store certain sensitive attributes as ciphertext, which prevents data from being readable while it is stored in underlying database files, backup files, and exported LDIF files. Attribute encryption enables you to encrypt important data before it is written to the disk and to decrypt data when it is read from the disk.

> **Note:**
>
> The attribute encryption feature does not encrypt data that is retrieved over the LDAP protocol. Only data saved on the disk is encrypted.
>
> If an LDAP client reads (searches for) an entry with some encrypted attributes on the disk, then that client receives a decrypted entry and the values of the originally encrypted attributes are immediately readable without any decryption.

Attributes are not encrypted by default. You configure attribute encryption at the suffix level, which means that an attribute is encrypted at every entry in which it appears in the suffix. Thus, after an attribute is encrypted, every instance of that attribute is encrypted before it is stored in the database files. This in turn implies that all of the on-disk data for that specific attribute is encrypted.

Encryption is always reversible. Encrypted attributes are decrypted when returned through search requests. If you want to encrypt an attribute in an entire directory, then you must enable encryption for that attribute in every suffix or leave the suffix list empty.

> **Note:**
>
> Attribute encryption affects all data and index files associated with a suffix. These attributes are not changed (encrypted) until attribute encryption is activated. Existing attributes will remain unchanged.
>
> To apply encryption to all of the data, you must first make the configuration change, export the contents, and then re-import the contents.

Attribute encryption also enables you to export data to another database in an encrypted format. The purpose of attribute encryption is to protect sensitive data only when the data is being stored or exported.

Related Topics

Masking Attributes in the Audit Log

14.2 Understanding Attribute Encryption

Attributes are not encrypted by default. You need to configure attribute encryption at the suffix level. This indicates that an attribute is encrypted at every entry in which it appears in the suffix.

Oracle Unified Directory allows you to encrypt:

- Specific attribute types defined in a mandatory attribute types list.

> **Note:**
>
> You cannot encrypt some operational or internal attributes, such as `entryuuid`, `createTimestamp`, virtual attributes, or password attributes. For more information about attributes that are not supported for encryption, see Considerations for Attribute Encryption Usage.

- Only DB Local Backend (user back end).
- Attributes in all suffixes of all available DB Local Backends or, if listed, in some specific suffixes. For example:

  If suffixes are specified, then it should be root suffixes of a DB Local Backend, not a sub suffix. For example, if DB Local Backend has root suffix `dc=example,dc=com` then you cannot encrypt some attributes only in `ou=people,dc=example,dc=com`.

14.3 Understanding Encryption Algorithms

Oracle Unified Directory enables you to prevent unauthorized access to attributes of an entry stored on a disk using encryption algorithms.

An encryption algorithm is a set of mathematical rules or functions used for encrypting and decrypting data. These algorithms work in combination with a key to encrypt and decrypt data.

The attribute encryption feature supports a wide range of standard encryption algorithms.

You can configure the server to encrypt attributes using several encrypting schemes. The supported encryption schemes include:

- AES128
- AES256

> **Note:**
>
> Download the "Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files" if applicable. The download page is https://www.oracle.com/in/java/technologies/javase-jce-all-downloads.html

- Blowfish (128-bit key)
- Triple DES (168-bit key)
- RC4 (128-bit key)

The newly supported encryption schemes include:

- AES-128-GCM
- AES-192-GCM
- AES-256-GCM

> **Note:**
>
> Oracle recommends you to use a GCM based stronger encryption algorithm from the above list. Weaker algorithms are used for backward compatibility. When a weaker encryption algorithm is detected by OUD server, a warning message gets logged in the server logs.

14.3.1 Attribute Encryption Key

Oracle Unified Directory uses symmetric key for attribute encryption and decryption. It generates the required symmetric key whenever it is needed. Symmetric key is encrypted and securely stored in `cn=admin data` backend. This encryption is done using the server's public key. Hence, the stored encrypted symmetric key can only be decrypted using the server's private key.

14.4 Understanding Encryption in Index Keys

An attacker can also access sensitive data directly through index files. Therefore, it is imperative to encrypt the index keys corresponding to the encrypted attributes, to ensure that the attributes are fully protected.

Database encryption is partially compatible with indexing. The content of the index files that are normally derived from attribute values are also encrypted to prevent an attacker from recovering part or all of the encrypted data from an analysis of the indexes.

The server pre-encrypts all index keys before looking up an index for an encrypted attribute. This action has some effect on server performance for searches that use an encrypted index. However, limited performance impact should not prevent you from using an index.

Oracle Unified Directory enables you to use the following index types for an associated encrypted attribute:

- Equality
- Substring
- Approximate
- Presence

> **Note:**
>
> You must bear in mind that encryption techniques do not preserve the order of an index. Therefore, ordering indexes are not supported when attributes are encrypted.

Encryption is supported for DB Local Backend indexes only. Keys of the indexes are encrypted for an encrypted attribute.

14.5 Understanding Encryption in Replication Topology

Encryption in replication topology refers to encrypting data that is stored in replication server databases.

Learn about how encryption is supported in a replication topology from the following topics:

- Understanding Encryption in a Replication Server Database (or `changelog`)
- Understanding Attribute Encryption Key in Replication Topology
- Updating Servers from 11.1.2.2.0
- Using an ODSEE Gateway

14.5.1 Understanding Encryption in a Replication Server Database (or `changelog`)

Oracle Unified Directory supports encryption in a replication server database (also known as the *changelog*) and for `cn=changelog` (also known as the *external changelog* or the *retro-changelog)*. Oracle Unified Directory encrypts data on a replication server database the same way it does for a server database. No additional configuration is necessary. Enabling and disabling encryption, defining attributes for encryption, and defining suffixes for encryption is the same for either database.

If you perform an operation on a server that is part of a replicated topology, and if that change is associated with an encrypted attribute, then Oracle Unified Directory encrypts the data in the replication server's database (the *changelog*, which is readable from `cn=changelog`) using the same algorithm that is used for encryption in the server.

When Oracle Unified Directory accesses the retro-changelog (cn=changelog), which accesses the `changelog`, the retro-changelog always returns clear values. Encryption only occurs at rest; that is, on stable storage (hard disk).

14.5.2 Understanding Attribute Encryption Key in Replication Topology

The keys used for encryption are created, stored, and retrieved from `cn=admin data`. This suffix is replicated on any other server in the topology.

So, any server in the topology can decrypt any encrypted attribute and send it to its LDAP clients. Therefore, keys used for encryption or decryption algorithm are replicated throughout the entire topology because `cn=admin data` is replicated.

A stored symmetric key used in encryption can only be decrypted by the server using its private key. Hence, while renewing ads-truststore certificate, you must retain the same keys.

> **Note:**
>
> If you are using a gateway from Oracle Directory Server Enterprise Edition, see Using an ODSEE Gateway .

14.5.3 Updating Servers from 11.1.2.2.0

When updating version 11.1.2.2.0 replicated topology of servers to version 11.1.2.3.0, encryption does not occur in every replication server database until after all servers have been updated.

In addition, you must wait for the purge delay to expire to ensure there are no more sensible values in the changelog.

14.5.4 Using an ODSEE Gateway

Learn about using an ODSEE gateway in this topic. Oracle Directory Server Enterprise Edition allows some attributes to be encrypted in the back end, but not in the changelog.

Starting with Oracle Unified Directory version 11.1.2.3.0, if you are using a gateway from Oracle Directory Server Enterprise Edition, then you can configure that gateway like other servers in the Oracle Unified Directory topology.

Then, if changes sent from an Oracle Directory Server Enterprise Edition server through the replication gateway are associated with an encrypted attribute (defined by the configuration as with regular Oracle Unified Directory servers), then Oracle Unified Directory can encrypt that data and store it in the replication server database.

14.6 Considerations for Attribute Encryption Usage

Learn about the usage of Attribute Encryption, what happens when the attribute encryption configuration is modified and what needs to be done for encrypted attributes that are indexed.

Consider the following points below while implementing the attribute encryption feature:

- Attribute encryption provides increased data security, but it does have an impact on system performance. Consider using encryption only for the most sensitive attributes.
- When modifying the attribute encryption configuration, you must export your data, make the configuration changes, and then import the newly configured data to ensure that all configuration changes are taken into account without any information loss. If you fail to do so, then data that is already present in the back end on which no change occurred after the data encryption configuration change remains in clear or encrypted format as configured with the initial algorithm.
- Algorithm changes are supported. Modifying encryption on an indexed attribute requires that you rebuild the index associated with the encrypted attribute. This in turn impacts the performance. For more information about rebuilding indexes, see rebuild-index.
- For encrypted attributes that are indexed, it is required to maintain the consistency between indexes and the data encryption configuration. If you modify or update the configuration for encrypted attributes, then you must rebuild the indexes associated with the encrypted attribute. Failing to do so will log an error message in the error log file, which prompts you to rebuild the indexes because the configuration has changed. For more information about how to rebuild indexes, see rebuild-index.
- If you configure an attribute of RDN to be encrypted, then the values that appear in the DN will not be encrypted. Only values that are stored in the entry are encrypted.

  For example, consider the following entry:

  ```
  dn: uid=foo,dc=example,dc=com
  objectclass: inetorgperson
  objectclass: organizationalperson
  objectclass: person
  objectclass: top
  uid=foo
  cn=bar
  sn=joe
  ```

  Here, `uid` is an attribute that is:

  – Part of the DN of the entry and is its RDN.

  - Also part of the attributes of the entry. You must keep in mind that this is always the case, because RDN is always present as an attribute in the entry.

  However, `uid` is a multi-valued attribute, therefore you can add a value to `uid` in the entry as follows:

```
dn: uid=foo,dc=example,dc=com
objectclass: inetorgperson
objectclass: organizationalperson
objectclass: person
objectclass: top
uid=foo
uid=secondValue
cn=bar
sn=joe
```

  Now, if you encrypt `uid`, then the new value that you have added is encrypted and not the initial value, `foo`. The value that is in the RDN is not encrypted.

- You cannot configure encryption for the following attributes because they are used internally by the server:

  Operational Attributes

  - `objectclass`
  - `entryUUID`
  - `creatorsName`
  - `createTimestamp`
  - `modifiersName`
  - `modifyTimestamp`

  Virtual Attributes

  You cannot configure a virtual attribute for encryption.

  Password Attributes

  Because passwords are already hashed or encrypted, you cannot use the attribute encryption feature to modify the existing behavior of, or configure encryption for, any password attributes that are defined in a password policy. For example, the `userPassword` attribute, which is defined in the default password policy is not supported.

  Password encryption or hashing is handled differently. For information about password policies and the password storage scheme, see Managing Password Policies.

14.7 Configuring Attribute Encryption

You must configure different parameters to enable attribute encryption.

The following sections describe the configuration parameters to enable attribute encryption and the different methods to configure attribute encryption.

- Attribute Encryption Configuration Parameters
- Attribute Encryption Advanced Configuration Parameters
- Configuring Attribute Encryption Using the `dsconfig` Command
- Configuring Attribute Encryption Using the `dsconfig` Interactive Mode
- Managing Attribute Encryption

14.7.1 Attribute Encryption Configuration Parameters

Learn about the various attribute encryption configuration parameters, their names, descriptions, allowable values and formats, and rules to perform an action based on the values provided, from the tabular column below.

Table 14-1 describes the configuration parameters to enable attribute encryption.

Table 14-1 Configuration Parameters for Attribute Encryption

Name	Description	Single/ Multi Valued	Format	Presence Rules
`enabled`	Allows you to enable or disable encryption.	`S`	String representing a boolean, true or false	If set to `true`, then you must at least define `attribute-encryption-include`.
`attribute-encryption-include`	Encrypts every attribute defined here. Encrypt attributes of all the entries of all suffixes or only in the suffixes defined with `encrypted-suffix` if defined.	`M`	String representing a single attribute name or OID	Defined if `enabled` is set to `true`
`encrypted-suffix`	Controls how encryption is applied for suffixes. • If not present, encryption is applied on suffixes stored in DB Local Backend. • If present, defines the list of user DB Local Backend suffixes on which the encryption is applied. Other suffixes are not encrypted. **WARNING:** The suffix must be a root suffix defined in the back end, not a descendant. For example, if back end has `dc=example,dc=com` defined as a supported suffix, you cannot use `ou=people,dc=example,dc=com` here.	`M`	String representing a single suffix	Meaningful if `enabled` is set to `true`

Table 14-1 (Cont.) Configuration Parameters for Attribute Encryption

Name	Description	Single/ Multi Valued	Format	Presence Rules
`attribute-encryption-algorithm`	Defines the algorithm to use for encryption.	`S`	String representing an encryption algorithm. Possible values are: • triple-des-168 • aes-128 • aes-256 • blowfish-128 • `rc4-128` • aes-128-gcm • aes-192-gcm • aes-256-gcm	Meaningful if `enabled` is set to `true`

Note:

The following algorithms: aes-128

Table 14-1 (Cont.) Configuration Parameters for Attribute Encryption

Name	Description	Single/ Multi Valued	Format	Presence Rules
				-gcm, aes-192-gcm, and aes-256-gcm are applicable for Oracle Wallet only.

Table 14-1 (Cont.) Configuration Parameters for Attribute Encryption

Name	Description	Single/ Multi Valued	Format	Presence Rules
				r21BundlePatchandlaterreleases.

14.7.2 Attribute Encryption Advanced Configuration Parameters

In this release, the following configuration parameters, except offline-re-encryption, is applicable for attribute encryption using scheduled tasks.

Table 14-2 Attribute Encryption Table

Name	Description	Single/ Multi Valued	Format	Presence Rules
`encryption-properties`	Optional encryption properties that you can use to configure custom values for initialization vector length in bits (`iv-length-bits=128`) and GCM tag length (`gcm-tag-length=12`). When you do not specify any value, OUD uses `96 bit iv` and a `tag length` of 16 for GCM if chosen.	M	String represents a single property with name and value. For example, `iv-length-bits=96`	Meaningful for encryption algorithm (like GCM) that supports `iv` and `tag length`
`use-defined-enc-algo-in-replication`	Controls the encryption algorithm that will be used to generate replication changelogs. In a cross-version topology with new OUD instances and old OUD instances, this value set to `false` (default setting) if older instances of OUD do not understand a newer algorithm. `True` specifies that if all instances of OUD are running the same version, a changelog is generated and published to other OUD instances using the defined encryption scheme.	S	String represents a boolean value, `true` or `false`.	

14.7.3 Configuring Attribute Encryption Using the `dsconfig` Command

You use the `dsconfig` command to configure attribute encryption.

Consider the scenario, where you plan to encrypt every attribute, `postalAddress` and `mail`, with `AES-128` algorithm in entries of user DB Local Backend root suffixes, `dc=customers,dc=com` and `dc=partners,dc=com`.

To configure attribute encryption using the `dsconfig` command:

1. Run the following commands sequentially.

   To configure attribute encryption for `postalAddress` attribute with `AES-128` algorithm in the `dc=customers,dc=com` suffix, run the following command:

   ```
   dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
   -j /local/password set-data-encryption-prop --set enabled:true \
   --set attribute-encryption-include:postalAddress \
   --set encryption-algorithm:aes-128 \
   --set encrypted-suffix:dc=customers,dc=com
   ```

   To add attribute encryption for `mail` attribute and to add encryption in the `dc=partners,dc=com` suffix, run the following command:

   ```
   dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
   -j /local/password \
   set-data-encryption-prop --add attribute-encryption-include:mail \
   --add encrypted-suffix:dc=partners,dc=com \
   ```

14.7.4 Configuring Attribute Encryption Using the `dsconfig` Interactive Mode

You can configure attribute encryption using the `dsconfig` command-line interactive mode.

Introduction of a Data Encryption subsection, located under the main Security menu, allows you to modify all of the configuration attributes described in Table 14-1.

Consider the following example to configure attribute encryption using the `dsconfig` command in interactive mode.

```
Oracle Unified Directory Configuration Console Main Menu
What do you want to configure?

    1)  General Configuration            7)   Virtualization
    2)  Authentication and authorization 8)   Load Balancing
    3)  Schema                           9)   Distribution
    4)  Replication                      10)  Integration
    5)  Local Data Source                11)  Http
    6)  Remote Data Source

    q)  quit

Enter choice: 2
Authentication and authorization Management Menu

What would you like to do?

    1)  Access Control Group Plugin  8)  Password Policy Import
    2)  Access Control Handler       9)  Password Storage Scheme
    3)  Crypto Manager               10) Password Validator
    4)  Data Encryption              11) Root DN
    5)  Key Manager Provider         12) SASL Mechanism Handler
    6)  Password Generator           13) Trust Manager Provider
    7)  Password Policy

    b)  back
    q)  quit

Enter choice [b]: 4

Configure the Properties of Data Encryption
```

```
Property                          Value(s)
-----------------------------------------------------------------
1)  attribute-encryption-include  description, givenname, mobile
2)  enabled                       true
3)  encrypted-suffix              "dc=example,dc=com"
4)  encryption-algorithm          aes-128

?)  help
f)  finish - apply any changes to the Data Encryption
c)  cancel
q)  quit

Enter choice [f]: ?

Component name: Data Encryption

Data Encryption allows to configure attribute encryption.

Option Types:

 r -- Property value(s) are readable
 w -- Property value(s) are writable
 m -- The property is mandatory
 s -- The property is single-valued
 a -- Administrative action is required for changes to take effect

Property                      Options  Syntax
-------------------------------------------------
attribute-encryption-include  rw---    OID
enabled                       rw-s-    BOOLEAN
encrypted-suffix              rw---    DN
encryption-algorithm          rw-s-    ALGORITHM
-------------------------------------------------
```

14.7.5 Managing Attribute Encryption

You can enable and disable attribute encryption, and modify and fetch attributes using *dsconfig* command.

The following topics describe how to enable and disable attribute encryption:

- Enabling Encryption for Attributes of Specific Suffixes
- Disabling Encryption
- Enabling Encryption for a Specific Attribute Using an Algorithm
- Modifying Attributes
- Fetching Attributes

14.7.5.1 Enabling Encryption for Attributes of Specific Suffixes

This section describes a scenario to encrypt every attribute, `postalAddress` and `mail`, with `3DES-168` algorithm in entries of user DB Local Backend root suffixes, `dc=customers,dc=com` and `dc=partners,dc=com`.

To configure attribute encryption for `postalAddress` use the following command:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
-j /local/password \
```

```
set-data-encryption-prop --set enabled:true \
--set encryptedsuffix:dc=customers,dc=com \
--set attribute-encryption-include:postalAddress \
--set encryption-algorithm:triple-des-168 \
```

To configure attribute encryption for `mail` use the following command:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
-j /local/password \
set-data-encryption-prop --add attribute-encryption-include:mail \
--add encrypted-suffix:dc=partners,dc=com \
```

You can configure attributes using the `set-data-encryption-prop` option of `dsconfig` command. For more information about the encryption parameters, see dsconfig.

In this example, configure encryption using the preceding two `dsconfig` commands sequentially. For more information, see Configuring Attribute Encryption Using the `dsconfig` Command.

14.7.5.2 Disabling Encryption

Use the following `dsconfig` command to disable encryption:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
-j /local/password \
set-data-encryption-prop --set enabled:false \
```

14.7.5.3 Enabling Encryption for a Specific Attribute Using an Algorithm

Use the following command to encrypt the `mobile` attribute with the `AES-128` algorithm:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" \
-j /local/password set-data-encryption-prop --set enabled:true \
--set attribute-encryption-include:mobile \
--set encryption-algorithm:aes-128 \
```

14.7.5.4 Modifying Attributes

You can modify the attributes through the `dsconfig` command with the `set-data-encryption-prop` subcommand as follows:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" /
-j /local/password set-data-encryption-prop --set "enabled:true"
```

> **Note:**
>
> Run the `dsconfig set-data-encryption-prop --help` command to explore the `set-data-encryption-prop` command option. For more information, see dsconfig.

14.7.5.5 Fetching Attributes

You can read the attributes through the `dsconfig` command with the `get-data-encryption-prop` subcommand as follows:

```
dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager" /
-j /local/password get-data-encryption-prop
```

```
Property                      : Value(s)
------------------------------:-------------------------------
attribute-encryption-include  : description, givenname, mobile
enabled                       : true
encrypted-suffix              : "dc=example,dc=com"
encryption-algorithm          : aes-128
```

14.8 Configuring Attribute Encryption in Replication Enabled Topology

You can initialize OUD topology with attribute encryption. The keys used for encryption are not stored in the keystore instead in cn=admin data suffix. This suffix is replicated across all the servers of the topology. Therefore, any OUD instance can decrypt encrypted attribute prior to sending it back to client application.

> **Note:**
>
> Consider a Replication Enabled Topology with attribute encryption already configured and the attribute data of all entries stored in an encrypted state in the topology. To renew or regenerate Self-Signed Certificates (for example, when the Certificates are about to expire), see Regenerating Certificates Using dsreplication regenerate-cert. To renew or reset Custom or CA Signed Certificates, see Providing Certificates Using dsreplication set-cert. The commands (`dsreplication regenerate-cert` and `dsreplication set-cert`) automate the process of resetting the Certificates in the replication topology.

14.8.1 Configuring Attribute Encryption in a New Replicated Topology Setup

You can configure the attribute encryption in an entirely new replicated topology setup.

Follow these steps:

1. Configure attribute encryption for all instances using dsconfig. See Configuring Attribute Encryption Using the dsconfig Command and Configuring Attribute Encryption Using the dsconfig Interactive Mode.
2. Enable Replication for all instances. See Enabling Replication Between Two Servers With dsreplication.

> **Note:**
>
> Scheduled re-encrypt task is not applicable for this new replication topology setup assuming no data present.

14.8.2 Configuring Attribute Encryption in the Existing Replicated Topology Setup

Consider an OUD instance running in a replicated topology. To secure value of attributes while storing the attributes on the disk, you need to define attribute encryption. The attribute data of

the newly created entries is encrypted in the replicated topology, whereas the attribute data of the existing entries are not be stored securely.

To apply encryption to all entries or the attribute data, you need to perform following steps.

Follow these steps:

1. Configure attribute encryption for all instances using dsconfig. See Configuring Attribute Encryption Using the dsconfig Command and Configuring Attribute Encryption Using the dsconfig Interactive Mode.
2. Perform encryption of the existing data as described in the section Encryption or Re-encryption of Existing Data.

14.9 Encryption or Re-encryption of Existing Data

In the existing environment, when attribute encryption is newly enabled or any existing attribute encryption related configuration such as the encryption algorithm changes, then encryption of the existing data present in the backend as per new encryption configuration needs to be performed.

You can perform the attribute encryption in different ways as follows:

- Encryption or Re-encryption of Existing Data Using Scheduled Task
- Encryption or Re-encryption of Existing Data Using Replication or Import
- Encryption or Re-encryption of Existing Data on Single Instance Using Export or Import

14.9.1 Encryption or Re-encryption of Existing Data Using Scheduled Task

You can perform the encryption of the existing data present in the backend by configuring a re-encrypt scheduled task on an OUD server instance. The scheduled task encrypts the required data on the given OUD server instance where it is run.

In case of a replicated topology, it also publishes necessary attribute encryption modify operations to other nodes over replication so that data on the other nodes get encrypted.

Follow these steps

1. Ensure that necessary attribute encryption configurations are present on all OUD instances in case of a replicated topology.
2. Take backup of all OUD instances.
3. Configure the scheduled task for re-encryption using command, `dsconfig create-task --type reencrypt`.

```
Recurring Task ./dsconfig create-task -h <OUD_INSTANCE> -p <ADMIN_PORT> -D
"cn=Directory Manager" -j /local/password -X --type reencrypt --backendID
userRoot --recurringTask "* * * * *" --batch-size 10
Schedule Task ./dsconfig create-task -h <OUD_INSTANCE> -p <ADMIN_PORT> -D
"cn=Directory Manager" -j /local/password -X --type reencrypt --backendID
userRoot --start 0
```

In this approach the existing data is encrypted without the downtime of OUD instances.

Consider that the backend has huge number of entries, then you can schedule a ReEncrypt Recurring Task. The reencryption is preferred to be performed in batches at the configured

intervals as per the schedule. Consider that the backend has fewer entries, then you can schedule a single time Scheduled Task. All entries are intended to be reencrypted at once as part of a single task.

Oracle recommends that you configure the task on the OUD instance such that no LDAP queries from any clients are processed until the re-encryption is finished. This instance should be included in the replication topology but not in the load balancer. This can be accomplished in one of following ways:

1. When you remove one node from the existing load balancer.
2. When you build a new instance specifically for this purpose.

14.9.1.1 Scheduled Task for Re-encryption

The scheduled task for re-encryption takes care of re-encrypting all entries based on parameters passed. On successful completion of the task, the status is marked as "Completed Successfully and stopped Recurring Task" for a Recurring Task or "Completed successfully" for a Scheduled Task.

Usage

```
dsconfig create-task {options}
```

For global options, see `dsconfig --help`.

> **Note:**
>
> Interactive mode for dsconfig create-task is not supported now. Hence use complete command as listed below.

Subcommand Options

Table 14-3 Create Task Options

Option	Description
--type {type}	Specifies the type of task to be created. **Note:** Currently only reencrypt type is supported.

Table 14-3 (Cont.) Create Task Options

Option	Description
-t, --start {startTime}	Indicates the date and time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.
--recurringTask {schedulePattern}	Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time and date pattern.
--batch-size {batchsize}	Represents the batch size to be used for re-encryption.
--backendID {backendName}	Represents the backend ID for the backend to re-encrypt.
--completionNotify {emailAddress}	Represents the email address of a recipient to be notified when the task completes. You may specify this option more than once.
--errorNotify {emailAddress}	Represents the email address of a recipient to be notified if an error occurs when this task executes. You may specify this option more than once.
--dependency {taskID}	Represents the ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.
--failedDependencyAction {action}	Represents the action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

Examples

The examples of creating a recurring task and a scheduled task is as follows:

Creating a Recurring Task

```
./dsconfig create-task -h <OUD_INSTANCE> -p <ADMIN_PORT> -D "cn=Directory
Manager" -j /local/password -X --type reencrypt --backendID userRoot --
recurringTask "* * * * *" --batch-size 10
```

Creating a Scheduled Task

```
./dsconfig create-task -h <OUD_INSTANCE> -p <ADMIN_PORT> -D  "cn=Directory
Manager" -j /local/password -X --type reencrypt --backendID  userRoot --start
20210830121500
```

14.9.1.2 Managing Scheduled Tasks

You can view all re-encryption tasks using manage-tasks command:

```
./manage-tasks -h <OUD_INSTANCE> -p <ADMIN_PORT> -D "cn=Directory Manager" -
j /local/password -n -s
```

Output

```
20210830043148359 ReEncrypt Completed successfully
ReEncryptTask-5f73a72c-7725-473f-89e8-3bc05f847d64 ReEncrypt Completed
Successfully and stopped Recurring Task
ReEncryptTask-5f73a72c-7725-473f-89e8-3bc05f847d64-20210830043300000
ReEncrypt Completed successfully
ReEncryptTask-5f73a72c-7725-473f-89e8-3bc05f847d64-20210830043600000
ReEncrypt Completed successfully
```

Troubleshooting Using Manage-Tasks

You can troubleshoot by using the `manage-tasks` command. Use the `manage-tasks` command to check if the scheduled Re-encrypt Tasks had any errors during re-encryption. The status of the tasks would either be in "Completed with errors and stopped Recurring Task" for a Recurring Task or in "Completed with errors" for a Scheduled Task.

Example

You can view all re-encryption tasks by using the `manage-tasks` command:

```
./manage-tasks -h <OUD_INSTANCE> -p <ADMIN_PORT> -D "cn=Directory Manager" -
j /local/password -n -s
```

Output

```
ReEncryptTask-7ec1bab7-d971-44bb-b780-ea5c803dcd4f ReEncrypt Completed with
errors and stopped Recurring Task
ReEncryptTask-7ec1bab7-d971-44bb-b780-ea5c803dcd4f-20210830032400000
ReEncrypt Completed with errors
ReEncryptTask-7ec1bab7-d971-44bb-b780-ea5c803dcd4f-20210830032500000
ReEncrypt Completed with errors
```

You can view more information using the following command:

```
./manage-tasks -h <OUD_INSTANCE> -p <ADMIN_PORT> -D  "cn=Directory Manager" -
j /local/password -n -i "ReEncryptTask-7ec1bab7-d971-44bb-b780-
ea5c803dcd4f-20210830032400000"
```

The above command displays the log messages for the task and the entries that have failed during the run:

```
ReEncrypt Options
------------------------------------------------------------------------
ds-task-reencrypt-failed-dn uid=user.1,ou=People,dc=example,dc=com
```

```
uid=user.5,ou=People,dc=example,dc=com
dc=example,dc=com:3,6,9
ds-task-reencrypt-backend-id userRoot
objectclass ds-recurring-task
ds-tasktopds-task-reencrypt
ds-task-reencrypt-creation-time 20210830032400Z
ds-task-reencrypt-batch-size 10

Last Log Message
------------------------------------------------------------------------------
[30/Aug/2021:03:24:00 +0000] severity="NOTICE" msgCount=11 msgID=9896350
message="ReEncrypt task
ReEncryptTask-7ec1bab7-d971-44bb-b780-ea5c803dcd4f-20210830032400000 finished
execution"
```

In case of errors, it is recommended to look at the logs for root cause of the errors and re-schedule a new re-encryption task so that the failed entries are retried again for re-encryption.

If Reencryption (via Reencrypt Tasks) has been scheduled partially covering only few suffixes, a warning message gets logged in the server logs during server startup.

14.9.2 Encryption or Re-encryption of Existing Data Using Replication or Import

This approach requires OUD instance downtime to encrypt the existing data in a replicated topology. To apply encryption to all entries or the attribute data using replication, you need to perform the following steps.

1. Perform a pre-external-initialization on any one of the instances. See dsreplication.
2. Perform an off-line import either on all instances or on one instance, and then perform Binary copy to the other instances or dsreplication initialize. See import-ldif.
3. Perform a post-external-initialization on any one of the instances. See dsreplication.

14.9.3 Encryption or Re-encryption of Existing Data on Single Instance Using Export or Import

You can perform the encryption or re-encryption of existing data on single instance using export or import in a non-replicated topology.

1. Do one of the following:
   - If you want the existing data present in the back end to be configured for encryption, export the data using the LDIF script:

     ```
     export-ldif -n userRoot -l /data/export.ldif
     ```

     For more information about exporting to LDIF, see export-ldif
   - If you only want future modifications to consider the new encryption configuration, go to Step 4.
2. Perform the following steps to re-import data, and stop.

a. Stop the server.

```
stop-ds
```

b. Import data.

```
import-ldif -n userRoot -l /data/export.ldif
```

For more information about importing from command line, see import-ldif.

> **Note:**
>
> Irrespective of the fact whether data is encrypted or not in the imported LDIF file, the `import-ldif` command saves the data in the back end as stated by the current server configuration. So, the import process allows you to encrypt or decrypt data as needed. For example, importing encrypted data in a server configured with no encryption allows you to store data unencrypted. In addition, if you import a clear LDIF file into a server configured for encryption, then it allows you to store data encrypted.
>
> The algorithm of the current configuration is always used. If you import an AES128 encrypted data set into the server with RC4 encryption configured, then data is re-encrypted and stored with RC4 configuration.

c. Start the server.

```
start-ds
```

When you import data, then it also builds the indexes. Therefore, there is no need to perform step 4.

3. Rebuild indexes.
If the suffix on which you want to configure encryption contains indexes for the impacted attributes, then rebuild those indexes. Run the following commands:

For example, if there are some indexes associated with the `postalAddress` attribute, then rebuild index using the following command:

```
rebuild-index -b dc=customers,dc=com -i postalAddress
```

Similarly, if there are some indexes associated with the `mail` attribute, then rebuild index using the following command:

```
rebuild-index -b dc=customers,dc=com -i mail
```

For more information about rebuilding indexes, see rebuild-index.

14.10 Use Case Scenarios

You may have some replicated topology which may impact if attribute encryption is used, especially when certain releases of OUD contain newer encryption schemes. Consider the use case scenarios as follows:

- Mixed Replicated Topology Containing Multiple Versions of OUD Instances
- Replicated Topology with all OUD Instances Support Configured Encryption Scheme

.

14.10.1 Mixed Replicated Topology Containing Multiple Versions of OUD Instances

A mixed version topology consists of the latest version of OUD, which may contain newer encryption schemes which are not available in older version of OUD. In such a mixed version topology containing multiple versions of OUD, it is recommended that newer encryption schemes that are released in the latest product should be configured after all OUD instances are moved to the latest version. However, you can choose a combination of newer and older encryption schemes in such a mixed version topology.

This requires the value of the configuration property, `use-defined-enc-algo-in-replication` to be set to `false`. With this configuration, latest release of OUD server instance will generate changelog or modify events using AES-128 encryption algorithm which is understood by all older versions of OUD server.

For example, consider a scenario where OUD1 instance is created using an older version with data encryption algorithm AES-256 and OUD2 instance is created using a newer version of OUD with data encryption algorithm AES-256-GCM. In this topology OUD2 understands newly introduced encryption algorithm whereas OUD1 fails to decipher the newer encryption algorithms. Now, any change in OUD2 will generate the change-log events using AES-256-GCM algorithm and same is replicated to OUD1. OUD1 fail to decrypt the change-log that is encrypted using AES-256-GCM.

To avoid such compatibility issues in mixed mode change-log is encrypted using the default AES-128 algorithm provided the configuration property, `use-defined-enc-algo-in-replication` is set to `false`. Once all instances are moved to the latest version, the configuration property, `use-defined-enc-algo-in-replication` is set to `true` so that replication will use the latest configured encryption algorithm.

14.10.2 Replicated Topology with all OUD Instances Support Configured Encryption Scheme

The replicated topology may contain different versions of OUD. However, in this topology, all OUD instances support the configured encryption scheme. Administrator can set the value of the configuration property, `use-defined-enc-algo-in-replication` to `true` so that any change in encrypted attribute values will generate the modify or change-log that is encrypted using configured encryption algorithm. This change-log is interpreted by other OUD node in the replication topology and replay the same change.

.

Part III
Basic Administration

This part describes how to start and stop the server and how to configure the various server elements, depending on the required deployment scenario.

This part includes the following chapters:

- Starting and Stopping the Server
- Accessing Oracle Unified Directory Using OUDSM
- Configuring the Server Instance
- Managing Directory Data
- Managing Users and Groups

15
Starting and Stopping the Server

Follow these topics which describe the basic procedures that apply to an Oracle Unified Directory directory server, proxy server, and replication gateway instance:

- Starting the Server
- Stopping the Server
- Checking the Server Status
- Running the Server as a Non-Root User
- Starting and Stopping Oracle Unified Directory Instance Created Within the Domain

15.1 Starting the Server

To start the server, run the `start-ds` command on UNIX or Linux systems or the `start-ds.bat` command on Windows systems. By default, the `start-ds` command starts the server as a background process when no options are specified.

You can use the `start-ds` command with the `--nodetach` option to run the server as a foreground process. For more information, see start-ds.

The `start-ds` command automatically attempts to find the correct Java environment to use when starting the server. You can specify the path to the Java installation, and provide additional options directly to the JVM when the directory server is starting. For more information, see Configuring the Default JVM and Java Arguments.

> **Note:**
>
> During server startup, if any OUD keystore is using weaker keysize or key algorithm or signature algorithm, then the following warning message appears in the server startup logs. For more information to update any certificate, see Managing Certificates Using dsreplication.
>
> ```
> "Certificate [ads-cert] is using weaker signature algorithm
> [SHA1withRSA] in keystore [asinst1/OUD/config/ads-keystore].It's
> recommended to use a stronger signature algorithm"
>
> "Certificate [ads-cert] is using weaker key bit size [1024] in
> keystore [asinst1/OUD/config/ads-keystore].It's recommended to use a
> stronger key bit size"
>
> "Certificate [ads-cert] is using weaker key algorithm [DSA] in
> keystore [asinst1/OUD/config/ads-keystore].It's recommended to use a
> stronger key algorithm"
> ```

The topics in this section include:

- Starting the Server Using `start-ds`
- Starting the Server as a Foreground Process
- Restarting the Server
- Starting the Server Using a Script (UNIX/Linux)

15.1.1 Starting the Server Using `start-ds`

You can start the server using `start-ds` command for Oracle Unified Directory (OUD) instances created with the following OUD setups: `oud-replication-gateway-setup`, `oud-proxy-setup` and `oud-setup`.

Follow the steps to start the server:

1. Change to the appropriate directory.

```
(UNIX, Linux)  $ cd INSTANCE_DIR/OUD/bin
(Windows)      C:\> cd INSTANCE_DIR\OUD\bat
```

2. Type `start-ds`.

```
(UNIX, Linux)  $ start-ds
(Windows)      C:\> start-ds
```

15.1.2 Starting the Server as a Foreground Process

You can start the server as a foreground process by changing to the appropriate directory and then by using the `start-ds` command.

To start the server as a foreground process:

1. Change to the appropriate directory.

```
(UNIX, Linux)  $ cd INSTANCE_DIR/OUD/bin
(Windows)      C:\> cd INSTANCE_DIR\OUD\bat
```

2. Type `start-ds` with `-N` or `--nodetach`.

```
(UNIX, Linux)  $ start-ds --nodetach
(Windows)      C:\> start-ds --nodetach
```

You can stop the directory server by pressing `Control-C` in the terminal window in which the server is running or by running the `stop-ds` utility from another window.

15.1.3 Restarting the Server

You can restart the server by changing to the installation directory and by using `--restart` command.

To restart the server:

1. Change to the installation directory.

```
(UNIX, Linux)  $ cd INSTANCE_DIR/OUD/bin
(Windows)      C:\> cd INSTANCE_DIR\OUD\bat
```

2. Type `stop-ds` with `-R` or `--restart`.

```
(UNIX, Linux)  $ stop-ds --restart
(Windows)      C:\> stop-ds --restart
```

15.1.4 Starting the Server Using a Script (UNIX/Linux)

The `start-ds` command provides a "quiet" option (`-Q` or `--quiet`) that suppresses output during the startup process unless a significant error occurs. You can use this option in a startup script.

To start the server using a script:

1. Create a shell script and add the following `start-ds` command.

```
INSTANCE_DIR/OUD/bin/start-ds --quiet
```

2. Run the script.

15.2 Stopping the Server

On any system (whether the server is running in the foreground or the background), or even from a remote system, you can stop the server using one of the following methods.

Follow the topics below to stop the server:

- Stopping the Server Using `stop-ds`
- Stopping the Server that is Running in the Foreground
- Stopping the Server Using a Script (UNIX/Linux)

For more information about the `stop-ds` command, see stop-ds.

15.2.1 Stopping the Server Using `stop-ds`

You can stop the server by changing to the appropriate directory and by using `stop-ds` command only for Oracle Unified Directory (OUD) instances created with the following OUD setups: `oud-replication-gateway-setup`, `oud-proxy-setup` and `oud-setup`.

To stop the server using `stop-ds` command:

1. Change to the appropriate directory.

```
(UNIX, Linux)  $ cd INSTANCE_DIR/OUD/bin
(Windows)        C:\> cd INSTANCE_DIR\OUD\bat
```

2. Type `stop-ds`

```
(UNIX, Linux)  $ stop-ds
(Windows)      C:\> stop-ds
```

15.2.2 Stopping the Server that is Running in the Foreground

You can stop the server that is running in the foreground by running the `stop-ds` command from another window.

This procedure assumes that the directory server is running as a foreground process (using the `-N` or `--nodetach` option).

- Type Control-C in a terminal window on UNIX or in the Command Prompt window on Windows systems to stop the server.

  Alternatively, run the `stop-ds` command from another window.

15.2.3 Stopping the Server Using a Script (UNIX/Linux)

The `stop-ds` command provides a "quiet" option (`-Q` or `--quiet`) that suppresses output during the stopping process unless a significant error occurs. You can use this option in a shutdown script.

To stop the server using a script:

1. Create a shell script and add the following `stop-ds` command.

   INSTANCE_DIR`/OUD/bin/stop-ds --quiet`

2. Run the script.

15.3 Checking the Server Status

You can check if the server is started or stopped at any time, by using the `status` command.

To check the server status:

1. Change to the appropriate directory.

```
(UNIX, Linux)  $ cd INSTANCE_DIR/OUD/bin
(Windows)      C:\> cd INSTANCE_DIR\OUD\bat
```

2. Type `status`

```
(UNIX, Linux)  $ status
(Windows)      C:\> status
```

15.4 Running the Server as a Non-Root User

Like many network daemons, Oracle Directory Server Enterprise Edition has a `setuid` capability that allows it to be started as a root user but then drop privileges to run as a user with fewer capabilities. Oracle Unified Directory does not currently include this capability. However, you can install, start, and run the server as a non-root user.

> **Note:**
>
> The information in this section applies primarily to UNIX-based platforms because Windows systems do not historically place as many restrictions on non-administrative users.

This section includes the following topics:

- Understanding the Rationale to Run the Server as a Non-Root User
- Running the Server as a Non-Root User on the Standard LDAP Ports

15.4.1 Understanding the Rationale to Run the Server as a Non-Root User

Often, running the server as a non-root user from the start is an option that provides greater functionality than the `setuid` equivalent. Running the server as a non-root user means that the administrators do not need root access to the system, which is often desirable from an operational perspective. In addition, more administrative actions can be performed with the server online, because the server can do things that might not have been available after it had dropped root privileges.

The primary reason that servers are typically started and run as root users is so that they can listen on a privileged port (namely, ports between 1 and 1024). The standard port for LDAP communication is port 389, and the standard port for LDAPS is 636. On most UNIX-based systems only root users can create processes that listen on these ports. There can be other reasons for starting as a root user (for example, the ability to use a larger number of file descriptors), but it is generally easier to configure around these other limitations.

Although the standard LDAP and LDAPS ports are 389 and 636, the server is not required to run on those ports. In some environments, it is common to run the server on ports above 1024 (such as 1389 and 1636) so that it is not necessary to be root to start it. Virtually all LDAP-enabled clients provide the ability to specify the port on which the server is listening. If the clients know which port the server is using, then any value is allowed. For information about configuring the listen port, see Displaying the Properties of LDAP Connection Handler.

15.4.2 Running the Server as a Non-Root User on the Standard LDAP Ports

If clients expect the server to be listening on port 389 or 636, other options are still available. The best option, available on Solaris systems from Solaris 10 onwards, is to use the process rights management subsystem (also called *least privilege*). The privileges subsystem in Solaris makes it possible to give non-root users and roles capabilities normally available only to the root user (much like the Privilege Subsystem allows within the server).

In particular, the `net_privaddr` privilege controls which users can bind to privileged ports. If this privilege is granted to a non-root user, that user can bind to privileged ports.

To configure a user with this privilege, run the following command, as the root user:

```
# usermod -K defaultpriv=basic,net_privaddr,sys_resource,-proc_info,-file_link_any oud
```

This command configures the `oud` user so that it starts with the `basic` privilege set (which is what non-root users have by default). The command then adds the `net_privaddr` and `sys_resource` privileges, which allow the user to increase the number of file descriptors available, among other things. The command removes the `proc_info` privilege (which allows the user to see processes owned by other users) and the `file_link_any` privilege (which allows the user to create hard links to files that they do not own). After running this command, the `oud` user can start the server listening on a privileged port.

Even on systems without a capability like least privilege, it is possible to expose the server on a privileged port such as 389 or 636 without requiring root privileges to be able to start it. One possibility would be to run the server on an unprivileged port and use a directory proxy server listening on the privileged port to forward communication to the server on an unprivileged port. It is also possible to use network hardware to achieve the same purpose or to use firewall rules on the same system.

For example, on Linux systems the following commands can be used to redirect traffic targeting port 389 to port 1389:

```
# iptables --append PREROUTING --table nat --protocol tcp --dport 389 \
  --jump REDIRECT --to-port 1389
# iptables -t nat -A OUTPUT -p tcp --dport 389 -j DNAT --to :1389
```

15.5 Starting and Stopping Oracle Unified Directory Instance Created Within the Domain

You can start and stop Oracle Unified Directory (OUD) instances using command line interface and WebLogic Scripting Tool (WLST) commands.

> **Note:**
>
> It is not recommended to start or stop an instance using `start-ds` or `stop-ds` commands when you have created the OUD instance within the domain. Use these commands only when you have created OUD instance with the following OUD setups: `oud-replication-gateway-setup`, `oud-proxy-setup` and `oud-setup`. Instances created within the domain must be started or stopped using either the `startComponent.sh` and `stopComponent.sh` commands from Command Line or using the WLST commands.

> **See Also:**
>
> - Setting Up the Directory Server Using the WebLogic Scripting Tool (WLST)
> - Setting Up the Proxy Server Using the WebLogic Scripting Tool (WLST)
> - Setting Up the Replication Gateway Using the WebLogic Scripting Tool (WLST)

This section covers the following topics:

- Starting Oracle Unified Directory Instance
- Stopping Oracle Unified Directory Instance

15.5.1 Starting Oracle Unified Directory Instance

You can start an Oracle Unified Directory (OUD) instance using command line interface and WebLogic Scripting Tool (WLST) commands..

> **Note:**
>
> You can run the start commands only on OUD instances created using WLST.

Before performing these actions, you need to start the NodeManager using the `./startNodeManager.sh` command as follows:

```
cd $DOMAIN_HOME/bin
Run ./startNodeManager.sh
```

This section contains the following topics:

- Starting Oracle Unified Directory Instance Using Command Line
- Starting Oracle Unified Directory Instance Using WebLogic Scripting Tool Commands

15.5.1.1 Starting Oracle Unified Directory Instance Using Command Line

You can start an Oracle Unified Directory (OUD) instance using command line interface.

To start an Instance:
Run the following command from command line interface to start an OUD Instance.

For example:

```
$DOMAIN_HOME/bin/startComponent.sh oud1
```

where `oud1` is the instance name/server name created using WLST

15.5.1.2 Starting Oracle Unified Directory Instance Using WebLogic Scripting Tool Commands

You can start an Oracle Unified Directory (OUD) Instance using WebLogic Scripting Tool (WLST) commands.

> **Note:**
>
> Before starting or stopping an instance, you need to connect to the NodeManager as follows:
> `nmConnect(domainName='base_domain',username='weblogic',password='password')`

To start an Instance:

Run the following WLST command to start an OUD Instance.

```
nmStart(serverName='oud1',serverType='OUD')
```

15.5.2 Stopping Oracle Unified Directory Instance

You can stop Oracle Unified Directory (OUD) instances using command line interface and WebLogic Scripting Tool (WLST) commands.

> **Note:**
>
> You can run the stop commands only on OUD instances created using WLST.

Before performing these actions, you need to start the NodeManager using the `./startNodeManager.sh` command as follows:

```
cd $DOMAIN_HOME/bin
Run ./startNodeManager.sh
```

This section contains the following topics:

- Stopping Oracle Unified Directory Instance Using Command Line
- Stopping Oracle Unified Directory Instance Using WebLogic Scripting Tool Commands

15.5.2.1 Stopping Oracle Unified Directory Instance Using Command Line

You can stop an Oracle Unified Directory (OUD) instance using command line interface.

To stop an Instance:
Run the following command from command line interface to stop an OUD Instance.

For Example:

```
$DOMAIN_HOME/bin/stopComponent.sh oud1
```

where `oud1` is the instance name/server name created using WLST

15.5.2.2 Stopping Oracle Unified Directory Instance Using WebLogic Scripting Tool Commands

You can stop an Oracle Unified Directory (OUD) Instance using WebLogic Scripting Tool (WLST) commands.

> **Note:**
>
> Before starting or stopping an instance, you need to connect to the NodeManager as follows:
> `nmConnect(domainName='base_domain',username='weblogic',password='password')`

To stop an Instance:

Run the following WLST command to stop an OUD Instance.

```
nmKill(serverName='oud1',serverType='OUD')
```

16

Accessing Oracle Unified Directory Using OUDSM

Oracle Unified Directory Services Manager (OUDSM) is an interface for managing instances of Oracle Unified Directory. OUDSM enables you to configure the structure of the directory, define objects in the directory, add and configure users, groups, and other entries. OUDSM is also the interface you use to manage entries, schema, security, and other directory features.
This chapter describes how to access Oracle Unified Directory by using OUDSM, and includes the following sections:

- Invoking OUDSM
- Connecting to the Server Using OUDSM
- Displaying Server Information Using OUDSM

Additional information about using OUDSM to manage Oracle Unified Directory is available in the following sections:

- Managing the Server Configuration Using OUDSM
- Searching Data Using OUDSM
- Managing Data Using OUDSM
- Managing Access Control Using OUDSM
- Managing Password Policies Using OUDSM
- Managing the Schema Using OUDSM

For information about using OUDSM to manage proxy configurations, see Configuring Proxy, Distribution, and Virtualization Functionality.

16.1 Invoking OUDSM

You can invoke OUDSM using a URL where you enter the host name and the port name as given in this topic.

To invoke OUDSM, enter the following URL into your browser's address field:

`http://host:port/oudsm`

where *host* is the name of the managed server on which OUDSM is running and *port* is the managed server port number of the admin server. The default admin port is 7001.

Connect to the server as described in Connecting to the Server Using OUDSM.

> **Note:**
>
> If Oracle Unified Directory has recently been updated, you might encounter a problem on Oracle WebLogic Server when you try to invoke OUDSM. During an Oracle Unified Directory update operation, OUDSM is also updated, and the OUDSM URL can change. This problem usually occurs if you used your browser to invoke the earlier version of OUDSM.
>
> Therefore, to invoke the updated version of OUDSM, first clear your browser's cache and cookies.

16.2 Connecting to the Server Using OUDSM

You can connect to a specific Oracle Unified Directory instance from the main OUDSM screen.

Enter the following information to connect to an Oracle Unified Directory instance:

- **Server.** Enter the name of the directory server to which you want to connect.
- **Port.** Enter the administration port number of the directory server to which you want to connect.
- **User name.** Enter the bind DN to connect to the directory.
- **Password.** Enter the bind password to connect to the directory.

If SSL is enabled, you are asked to trust the server certificate.

> **Note:**
>
> - If you change the browser language setting, then you must update the session to use the new setting. To update the session, either reenter the OUDSM URL in the URL field and press Enter or quit and restart the browser.
> - You can configure OUDSM to use protocols and cipher suites that the Oracle Unified Directory server supports for TLS communication. See Configuring TLS Protocol and Cipher Suites for OUDSM to OUD Communication.

16.3 Displaying Server Information Using OUDSM

The Home tab and Metrics tab of each server instance in OUDSM enables you to view specific information about the server.

This section describes how to view server information and contains the following topics:

- Understanding the Server Role
- About Version Information
- About Server Statistics
- About the Configured Connection Handlers
- About the Configured Naming Contexts
- About the Configured Data Sources

- About Server Metrics

16.3.1 Understanding the Server Role

The server role can be one or more of the following, depending on how the Oracle Unified Directory instance was set up.

- Directory
- Proxy
- Load Balancer
- Distributor
- Replication Gateway
- Replication Server

For more information, see Selecting a Server Role.

16.3.2 About Version Information

The version information panel indicates the version number of the OUDSM instance, the Oracle Unified Directory instance, and the version of the Java Runtime Edition (JRE).

16.3.3 About Server Statistics

The OUD Statistics panel displays installation details and basic monitoring information for this server instances.

The following information is displayed:

- **Server Start Time.** The latest date and time on which the server was started successfully.
- **Installation Path.** The network path to the installation files for this server instance.
- **Instance Path.** The network path to the instance files for this server instance.
- **Administrative Users.** The root user that was configured when the server was set up. For more information, see Managing Administrative Users.
- **Total LDAP Operations Completed (per sec) (since startup).** The total number of LDAP operations performed on the server, divided by the number of seconds that have passed since server startup.
- **Average Elapsed Time per Operation (since startup) (ms).** The average time taken to complete an LDAP operation.
- **Connection Rate (con/sec).** The number of connections that the server is currently handling per second.

16.3.4 About the Configured Connection Handlers

The Connection Handlers panel details of all the connection handlers that are configured for this server instance, including the type of connection handler, the port on which that connection handler is listening, and whether the connection handler is enabled.

For more information about connection handlers, see Configuring Connection Handlers Using `dsconfig` and Managing Administration Traffic to the Server.

16.3.5 About the Configured Naming Contexts

The Naming Contexts panel displays all naming contexts, or suffixes, that are configured on this server instance, including the network group to which that naming context belongs, the number of entries in the naming context and whether that naming context is replicated.

16.3.6 About the Configured Data Sources

For proxy servers, the Data Sources panel displays all of the data sources, or back-end LDAP servers that are managed through that proxy instance.

16.3.7 About Server Metrics

The Metrics tab of each server instance in OUDSM enables you to view specific performance metrics for the server. These metrics include the server start up time, its current time, and usage information.

By default, this page is configured to automatically refresh the displayed metrics every 20 seconds. To change the default refresh interval, click the **Update** link and enter a new value. You can disable this feature by clearing the **Automatic Refresh** box.

Icons indicate whether the metrics are up  , down  , or unchanged  since the last refresh. In addition, a Help icon  indicates that additional information is available. Click the Help icon  to view a pop-up with information related to that entry.

The usage information is organized into three subtabs, which are described in the following topics:

- About Usage Since Startup Display
- About Current Usage Display
- About Cache Usage Display

16.3.7.1 About Usage Since Startup Display

The Usage Since Startup tab metrics are cumulative and they represent the sum of usage metrics since the server started up.

These metrics are provided on a server basis and on a connection handler basis. In each section, the header row shows the total metrics for the entire server and the tables show how the metrics are distributed across the connection handlers (or client ports).

The Usage Since Startup tab contains the following sections:

- **Opened Connections**. Shows how many connections have been opened since startup, per connection handler.
- **Entries Sent to Clients**. Shows how many entries have been sent to the clients since startup, per connection handler.
- **Megabytes Sent to Clients**. Shows how many megabytes have been sent to clients since startup, per connection handler.
- **Operations Completed**. Shows how many operations have been completed since startup, per connection handler.

- **Operations Initiated**. Shows how many operations have been initiated since startup, per connection handler.

Expand the header row in each section to view a table with the following information:

Parameter	Description
Connection Handler Name	Name of the connection handler that is managing this connection.
Port	Server port that the connection is using.
Since Startup	Absolute metric value since the server was started.
Avg/Sec Since Startup	Average metric value since the server was started.
Avg/Sec Since Previous Refresh	Average metric value between two refreshes. **Note**: This value can be quite useful because it shows what is currently happening in the server. The two averages illustrate the evolution of a metric between refreshes, which enables you to see if things are going up or down.

Note:

- Click the Sort Ascending or Sort Descending icons to sort the information in each column.
- Click the Help icon to view more information about that entry.

16.3.7.2 About Current Usage Display

The Current Usage tab displays metrics that represent how the Oracle Unified Directory server is currently being used, such as how many connections are currently open and the number of threads in the Java process.

The Current Usage tab contains the following sections:

- **Active Threads**. Shows how many active threads are currently being used.
- **Databases in Use**. Shows how many local database workflow elements are currently being used.
- **Open Connections**. Shows how many connections are currently open.

  Expand the header row to view a table with the following information:

Parameter	Description
Time Opened	Time that the connection was opened.
Bound As	User (Bind DN) that the connection is using to bind to the server.
Type	Type of connection. Possible values are: – **LDAP** for non-encrypted LDAP connections – **LDAPS** for SSL LDAP connections – **Start TLS** for LDAP connections using Start TLS – **JMX** for JMX connections
Connection ID	Unique integer identifier that is assigned to the server connection.

Parameter	Description
Connection Handler	Name of the connection handler that is managing this connection.
Port	Server port that the connection is using.
Client Address	IP address of the client accessing the server.
Server Address	IP address of the server.
Operations in Progress	Number of operations that are currently being handled by the connection. The same connection can be used to perform several operations in parallel.

Note:

To sort the information in each table column, click the Sort Ascending or Sort Descending icons .

- **Remaining Available Connections**. Shows how many connections can still be opened when the number of connections to the server is limited.

  You can configure the server to limit the number of connections by using the `max-allowed-client-connections` attribute. For more information about this attribute, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

Note:

Icons indicate whether the metrics are up , down , or unchanged since the last refresh.

16.3.7.3 About Cache Usage Display

The Cache Usage tab displays metrics about the database caches (one for each local database workflow element that is defined in the server) and the entry cache.

The Cache Usage tab contains the following sections:

- **Database Cache**. Shows information about each database cache.

  This table in this section provides the following information for the upper, bottom, and leaf nodes in the database.

Parameter	Description
Local DB Workflow Element Name	Name of the element.
Hits	Number of hits since server startup.
Tries	Number of tries since server startup.
Hit Ratio Since Startup	Hit ratio since server startup.
Hit Ratio Since Previous Refresh	Hit ratio since the last server refresh (most relevant value).

> **Note:**
>
> – The higher the hit ratios, the better the database cache is being used.
>
> – Click the Help icon next to a Database Cache Name to see information about the naming contexts associated with the local database workflow element.

- **Entry Cache**. Shows information about each entry cache that is defined in the server.

  The table in this section displays the following information about the entry cache configuration and hit ratio.

Parameter	Description
Name	Name of the cache.
Type	Type of cache configuration. Possible values are: – **File System** – **FIFO**
Hits	Number of hits since server startup.
Tries	Number of tries since server startup.
Hit Ratio Since Startup	Hit ratio since server startup.
Hit Ratio Since Previous Refresh	Hit ratio since the last server refresh (most relevant value).
DB Cache Max Size	Maximum size that the file system entry cache can take. **Note**: This parameter is displayed only when the entry cache type is *File System*.
Max Entries	Maximum number of entries that the cache can use. **Note**: This parameter is displayed only when the entry cache type is *FIFO*.
Max JVM Memory Percentage	Maximum percentage of JVM memory that the cache can use.

The expected hit ratio depends on how you defined the cache; in particular on the type of entries contained in the cache.

> **Note:**
>
> - Icons indicate whether the metrics are up , down , or unchanged since the last refresh.
> - Click the Help icon next to an Entry Cache Name to see more information, such as the include and exclude filters that define which entries should go in the entry cache.

17
Configuring the Server Instance

Understand how to configure and manage a server instance using the `dsconfig` command or using Oracle Unified Directory Services Manager.
The topics in this section are:

- Managing the Server Configuration Using `dsconfig`
- Managing Suffixes Using `manage-suffix`
- Managing the Server Configuration Using OUDSM
- Managing Administration Traffic to the Server
- Configuring Commands As Tasks
- Deploying and Configuring the DSML Gateway
- Managing the OUDSM Session Timeout

17.1 Managing the Server Configuration Using `dsconfig`

You can use the `dsconfig` command to configure both the Oracle Unified Directory directory server and the proxy server.

For a list of the supported subcommands for the directory server or proxy instance, and for specific information about this command, see dsconfig. You can also use `dsconfig` to configure some proxy-specific components.

> **Note:**
>
> The topics in this section are intended for administrators or users who want to configure and manage a deployed Oracle Unified Directory instance. These topics provide an overview of the dsconfig command-line utility and its use in server configuration.

This section contains the following topics:

- Using the `dsconfig` Command
- Using `dsconfig` in Interactive Mode
- Getting Help With `dsconfig`
- Configuring a Server Instance Using `dsconfig`
- Configuring Connection Handlers Using `dsconfig`
- Configuring Network Groups Using `dsconfig`
- Configuring Workflows Using `dsconfig`
- Configuring Workflow Elements Using `dsconfig`

- Configuring Plug-Ins Using `dsconfig`
- Configuring Suffixes with `dsconfig`.
- Configuring Access Control Groups With `dsconfig`

17.1.1 Using the `dsconfig` Command

The `dsconfig` command provides a simple mechanism for accessing the server configuration. `dsconfig` presents the configuration as a set of components, each of which can be managed through one or more subcommands.

You can also use `dsconfig` interactively. In interactive mode, `dsconfig` functions much like a wizard, walking you through the server configuration. For more information, see Using `dsconfig` in Interactive Mode.

You can only use `dsconfig` to configure a *running* server instance. Offline configuration is not supported by `dsconfig`.

Like the other administration commands, `dsconfig` uses the administration connector to access the server. For more information, see Managing Administration Traffic to the Server. All of the examples in this section assume that the administration connector is listening on the default port (4444) and that the command is accessing the server running on the local host. If this is not the case, the `--port` and `--hostname` options must be specified.

This section contains the following topics:

- Running `dsconfig` and Certificate Checking
- Working with `dsconfig` Subcommands
- Working with `dsconfig` Advanced Properties

17.1.1.1 Running `dsconfig` and Certificate Checking

The `dsconfig` command accesses the server over a secured connection with certificate authentication. If you run `dsconfig` in interactive mode, then you are prompted about how you want to trust the certificate.

If you run `dsconfig` in non-interactive mode (that is, with the `-n` option), specification of the trust store parameters depends on whether you run the command locally or remotely.

This section contains the following topics:

- Running `dsconfig` locally
- Running `dsconfig` remotely

17.1.1.1.1 Running `dsconfig` locally

To run the command locally, you need to launch `dsconfig` on the server that you are administering. If you do not specify the trust store parameters, the server uses the local instance trust store by default. Unless you specify otherwise, the local instance trust is `INSTANCE_DIR/OUD/config/admin-truststore`.

17.1.1.1.2 Running `dsconfig` remotely

To run the command locally, you need to launch `dsconfig` on a different server to the one you are administering. You *must* specify the trust store parameters or the `-X` (`--trustAll`) option. The easiest way to specify the trust store parameters is to run the command once in interactive mode and to save the certificate that is presented by the server in your trust store.

```
$ dsconfig

>>>> >>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server hostname or IP address [host1.example.com]:
Directory server administration port number [4444]:
Administrator user bind DN [cn=Directory Manager]:
Password for user 'cn=Directory Manager':

How do you want to trust the server certificate?

    1)  Automatically trust
    2)  Use a truststore
    3)  Manually validate

Enter choice [3]: 3

Administrator user bind DN [cn=Directory Manager]:

Password for user 'cn=Directory Manager':

Server Certificate:

User DN  : CN=host1.example.com, O=Administration Connector Self-Signed Certificate
Validity : From 'Wed Apr 29 11:13:21 MEST 2009'
             To 'Fri Apr 29 11:13:21 MEST 2011'
Issuer   : CN=host1.example.com, O=Administration Connector Self-Signed Certificate

Do you trust this server certificate?

    1)  No
    2)  Yes, for this session only
    3)  Yes, also add it to a truststore
    4)  View certificate details

Enter choice [2]: 3

Truststore path: /local/instances/certificates/jctruststore

Password for keystore '/local/instances/certificates/jctruststore':

...
```

When you have saved the certificate in the trust store, you can specify those trust store parameters in non-interactive mode.

```
$ dsconfig -h localhost -p 4444 list-connection-handlers -n \
--trustStorePath /local/instances/certificates/jctruststore \
--trustStorePasswordFile /local/instances/certificates/jctruststore.pin -j pwd-file
Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
LDIF Connection Handler  : ldif : false   : -
```

17.1.1.2 Working with `dsconfig` Subcommands

The `dsconfig` command provides an intuitive list of subcommands to manage various elements of the configuration.

You can use these subcommands to add, delete, list, view, and modify different components:

Subcommand	Function
`dsconfig create-component options`	Creates a new component
`dsconfig delete-component options`	Deletes an existing component
`dsconfig get-component-prop options`	Displays the properties of a component
`dsconfig list-components options`	Lists the existing defined components
`dsconfig set-component-prop options`	Modifies the properties of a component

For example, the following five subcommands are used to manage connection handlers:

Subcommand	Function
`dsconfig create-connection-handler options`	Creates connection handlers
`dsconfig delete-connection-handler options`	Deletes connection handlers
`dsconfig get-connection-handler-prop options`	Displays the properties of a connection handler
`dsconfig list-connection-handlers options`	Lists the existing defined connection handlers
`dsconfig set-connection-handler-prop options`	Modifies the properties of a connection handler

Not all types of components can be created and deleted. For example, a directory server has only a single global configuration. For this reason, the global configuration is managed with only two subcommands:

Subcommand	Function
`dsconfig get-global-configuration-prop options`	Displays the global configuration properties
`dsconfig set-global-configuration-prop options`	Modifies the global configuration properties

The configurable properties of all components can be queried and modified to change the behavior of the component. For example, an LDAP connection has properties that determine its IP listener address, its port, and its SSL configuration.

17.1.1.3 Working with `dsconfig` Advanced Properties

Some component properties are considered *advanced* properties. These advanced properties are not displayed by default, and have default values that apply in most cases. If you want to modify the advanced properties or their values, use `--advanced` before the subcommand. For example:

```
$ dsconfig --advanced get-extension-prop
```

17.1.2 Using `dsconfig` in Interactive Mode

`dsconfig` runs in interactive mode, unless you specify all configuration parameters and the `-n` (`--no-prompt`) option. Interactive mode functions like a wizard, walking you through the server configuration. Interactive mode is a good approach to start using `dsconfig`.

When you run `dsconfig` in interactive mode, you can specify that you want the equivalent command (including all your selections) to be displayed or written to a file.

The following example shows how to use the `--displayCommand` option to display the equivalent non-interactive command when configuring the trust manager. Notice that the equivalent command is displayed at the point at which the command has been applied and validated by the directory server.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --displayCommand
...
The TrustStore Manager Provider was modified successfully

The equivalent non-interactive command-line is:
dsconfig --hostname "localhost" --port "4444" --bindDN "cn=directory
manager" --bindPasswordFile pwd-file --trustAll
set-trust-manager-provider-prop --provider-name "PKCS12" --set
"enabled:true"
```

To copy the equivalent command to a file, use the `--commandFilePath` option, as shown in the following example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --
commandFilePath /tmp/filename
```

17.1.3 Getting Help With `dsconfig`

The `dsconfig` command has extensive online help that is accessed using the `--help` option.

This section provides an overview, and contains the following topics:

- Displaying Global Usage
- Finding the Correct Subcommand
- Getting Help for an Individual Subcommand
- Displaying a Summary of a Component's Properties
- Displaying Detailed Help on a Property

17.1.3.1 Displaying Global Usage

To view the global usage of the dsconfig command, you must use the `--help` option.

For example, run the command as follows:

```
$ dsconfig --help
```

17.1.3.2 Finding the Correct Subcommand

The global usage information does not include the list of available subcommands. To retrieve the list of subcommands, use one of the `--help-xxx` options, where *xxx* determines the group of subcommands to be displayed.

> **Note:**
>
> Use the `--help-all` option used to display all of the available subcommands.

For example, to find all the subcommands relating to distribution, use the following command:

```
$ dsconfig --help-distribution
```

17.1.3.3 Getting Help for an Individual Subcommand

When you have determined which subcommand you want, you can get more detailed help on that subcommand using the subcommand `--help` option as follows:

```
$ dsconfig create-monitor-provider --help
```

17.1.3.4 Displaying a Summary of a Component's Properties

The `dsconfig` command has built-in documentation for all of the components and their properties. You can access this documentation using the `list-properties` subcommand. For example, a summary of the properties associated with a work queue can be displayed using the following command:

```
$ dsconfig list-properties -c work-queue
```

If the `-c` option is not specified, a summary of the properties for all components is displayed.

17.1.3.5 Displaying Detailed Help on a Property

The summary table displays only brief usage information for each property. More detailed information are available using the verbose mode of the `list-properties` subcommand:

```
$ dsconfig list-properties -c work-queue --property num-worker-threads -v
```

If the `--property` option is not specified, verbose help is provided for all the work-queue properties.

17.1.4 Configuring a Server Instance Using `dsconfig`

The `dsconfig` command is the recommended utility for accessing the server configuration. You are not encouraged to access the configuration directly over LDAP, using the `ldap*` utilities.

This section describes the utility to access the server components and contains the following topics:

- Viewing the Properties of a Component
- Listing Components

- Understanding How Server Changes Are Recorded
- Creating a Component
- Modifying Component Properties
- Modifying the Values of a Multi-Valued Property
- Deleting a Component
- Using `dsconfig` in Batch Mode

17.1.4.1 Viewing the Properties of a Component

You can use a component's `get-xxx-prop` subcommand to view a list of its properties. Each component is associated with a single LDAP entry in the server configuration, and each property is associated with a single LDAP attribute.

To display the properties of the default LDAP connection handler, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-connection-handler-prop --handler-name "LDAP Connection Handler"

Property : Value(s)
-----------------------:-------------------------------------------------------
allow-ldap-v2 : true
allow-start-tls : false
allowed-client : -
denied-client : -
enabled : true
keep-stats : true
key-manager-provider : -
listen-address : 0.0.0.0
listen-port : 1389
ssl-cert-nickname : server-cert
ssl-cipher-suite : -
ssl-client-auth-policy : optional
ssl-protocol : -
trust-manager-provider : -
use-ssl : false
```

The `dsconfig` command displays the default values or behavior for properties that have not been customized.

17.1.4.2 Listing Components

You can view a list and summary of the instances of one component using the component's `list-`*xxx*`s` subcommand. This can be particularly useful if you have more than one instance of the same component.

For example, to list the configured connection handlers, run this command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-connection-handlers
```

Depending on your installation, the output will be similar to the following.

```
Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
```

```
LDIF Connection Handler  : ldif : false   : -          : -
SNMP Connection Handler  : snmp : false   : 161        : -
```

17.1.4.3 Understanding How Server Changes Are Recorded

Whenever someone makes a change to the server (`ADD`, `MODIFY`, `DELETE`, etc.), Oracle Unified Directory stores that change as an entry containing information; including which object was changed, which attributes were changed, and who made the changes.

This section contains the following topics:

- Overview of How Server Changes are Recorded
- Configuring How Server Changes are Recorded

17.1.4.3.1 Overview of How Server Changes are Recorded

The server itself automatically generates and handles either the `modifiersName` attribute or the `creatorsName` attribute, as follows:

- For MODIFYs and DELETEs, the server creates the `modifiersName` attribute.
- For ADDs, the server creates the `creatorsName` attribute.

Server changes can be explicitly performed by one user (user1) or by a user (user1) acting as another user (user2).

- If a single user (*user1*) performs the change, then there is no ambiguity and that modifiers's name or creator's name is stored.
- If a user (*user1*) performs the change acting as another user (*user2*), then *user1* binds to the server, but "becomes" *user2* to modify the object.

17.1.4.3.2 Configuring How Server Changes are Recorded

You can choose how you want the server to record these changes by configuring the `use-authid-for-audit-attrs` attribute. For example,

- **False** (*default*): Stores the authentication ID, such as the bind DN, of the bound user (*user1*) as the modifier.
- **True**: Stores the authorization ID of the proxied user (*user2*) as the modifier (If relevant, for example, when using proxy auth). The server records the authorization ID in the `creatorsName` or `modifiersName` during a write operation on the entry.

The following example illustrates setting the `use-authid-for-audit-attrs` attribute value to **`true`**, so that the server will record the proxied user (*user2*) as the modifier:

```
./dsconfig set-plugin-prop \
   --plugin-name LastMod \
   --set use-authid-for-audit-attrs:true \
   --hostname localhost \
   --port 4444 \
   --trustAll \
   --bindDN cn=Directory\ Manager \
   --bindPasswordFile /tmp/dsconfigpwd \
   --no-prompt
```

Related Topic

proxied authorization control

17.1.4.4 Creating a Component

You can create new instances of a component using the component's `create-xxx` subcommand.

Components often have several *subtypes*. For example, there are four types of connection handler: LDAP, LDIF, JMX, and SNMP. Because all of these are created using the same subcommand, you must specify the type of component that you want to create using the `-t` or `--type` subcommand.

When you create a new component, you must specify the component's mandatory properties. The mandatory properties depend on the type of component that is being created. For example, an LDAP connection handler might have different mandatory properties to a JMX connection handler. If a mandatory property is left undefined, `dsconfig` enters interactive mode and prompts you for the undefined properties. If you include the `-n` (non-interactive) option, `dsconfig` fails to create the component and displays an error message indicating which properties need to be defined.

1. Display the types of connection handler that can be created by accessing the help for the connection handler component.

```
$ dsconfig create-connection-handler --help

Usage: dsconfig create-connection-handler {options}
Creates Connection Handlers

Global Options:
See "dsconfig --help"

SubCommand Options:
--handler-name {NAME}
The name of the new Connection Handler
--set {PROP:VALUE}
Assigns a value to a property where PROP is the name of the property and
VAL is the single value to be assigned. Specify the same property multiple
times in order to assign more than one value to it
-t, --type {TYPE}
The type of Connection Handler which should be created. The value for TYPE
can be one of: custom | jmx | ldap | ldif | snmp
```

2. Create a new LDAP connection handler, specifying values for the mandatory `enabled` and the `listen-port` properties.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-connection-handler -t ldap --handler-name "My LDAP Connection Handler"
```

An error message similar to the following will be displayed.

```
The LDAP Connection Handler could not be created because the following
mandatory properties were not defined:

Property     Syntax
----------------------------------
enabled      false | true
listen-port  1 <= INTEGER <= 65535
```

17.1.4.5 Modifying Component Properties

The properties of a component can be modified using the component's `set-xxx-prop` subcommand. Multiple properties can be modified at the same time using multiple occurrences of the `--set` option. The following example uses the `set-connection-handler-prop` subcommand to modify the properties of a connection handler.

> **Note:**
>
> Many components have a Java class property that specifies the name of a Java class to be used as the implementation of the component. Do not modify this property, as doing so could prevent your server from operating correctly. These properties are treated as *advanced* properties and hidden from view unless you run `dsconfig` with the `--advanced` option.

For example, to configure the LDAP connection handler to accept LDAPv2 connections, run this command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name="LDAP Connection Handler" \
  --set allow-ldap-v2:true
```

17.1.4.6 Modifying the Values of a Multi-Valued Property

You can set multiple values for a property using the `--add` option multiple times in the same `dsconfig` command.

This example sets multiple values for the `allowed-client` property.

To restrict connections through the LDAP connection handler to specific clients, run these commands:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --add allowed-client:myhost --add allowed-client:myhost.example \
  --add allowed-client:myhost.example.com
```

17.1.4.7 Deleting a Component

Existing instances of a component can be removed using the `dsconfig delete-xxx` subcommand

The following example deletes the LDAP connection handler that was created in the previous example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  delete-connection-handler --handler-name "My LDAP Connection Handler"
```

17.1.4.8 Using `dsconfig` in Batch Mode

You can use the `-F` or `--batchFilePath` option of the `dsconfig` command to specify operations that are completed in a single command by consolidating those operations in a file. Consolidating these operations can significantly improve performance when several `dsconfig` commands are required.

To use `dsconfig` in batch mode, complete the following steps:

1. Create a script that contains the required commands for creating a new back end that is used to store a new suffix.

   For example, the following file (named `new-backend.txt`) achieves three separate tasks:

   - Creates the db-local-backend workflow element
   - Adds a set of index entry limit for the uniquemember attribute (for example, how to set properties, but this step is not mandatory)
   - Creates the workflow for the new suffix
   - Registers the new suffix in the default network group

   ```
   create-workflow-element --element-name myBackend --type db-local-backend \
   --set enabled:true --set base-dn:cn=myexample,cn=com
   set-local-db-index-prop --element-name myBackend --index-name uniqueMember \
   --set index-entry-limit:5000
   create-workflow --workflow-name myWorkflow --set base-dn:cn=myexample,cn=com \
   --set enabled:true --set workflow-element:myBackend
   set-network-group-prop --group-name network-group --add workflow:myWorkflow
   ```

2. Run the `dsconfig` command with that file as a parameter.

   ```
   $ dsconfig -h localhost -p 4444 -D cn="directory manager" -j pwd-file \
     -F new-backend.txt -X -n
   ```

17.1.5 Configuring Connection Handlers Using `dsconfig`

Connection handlers are responsible for handling all interaction with client applications, including accepting connections, reading requests, and sending responses.

For information about configuring secure connections, see Configuring SSL and StartTLS for LDAP and JMX.

The section describes how to configure the connection handlers using the `dsconfig` command, and contains the following topics:

> **Note:**
>
> The topics discussed in this section provide examples on only a few aspects of the configuration. For details about all the configuration properties, use the following command:
>
> ```
> $ dsconfig list-properties -c connection-handler
> ```

- Understanding Connection Handlers
- Displaying the Properties of LDAP Connection Handler
- Controlling Client LDAP Access to the Directory Server
- Configuring the LDIF Connection Handler
- Configuring the JMX Connection Handler

17.1.5.1 Understanding Connection Handlers

Oracle Unified Directory supports several types of connection handlers. It is important to understand each type and to learn how to display them.

This section contains the following topics:

- Understanding Types of Connection Handlers
- Viewing All Connection Handlers

17.1.5.1.1 Understanding Types of Connection Handlers

Oracle Unified Directory supports the following types of connection handlers:

- **LDAP connection handler.** This connection handler is used to interact with clients using LDAP. It provides full support for LDAPv3 and limited support for LDAPv2.
- **LDAPS connection handler.** This connection handler is used to interact with clients using LDAP over SSL.
- **LDIF connection handler.** This connection handler is used to process changes in the server using internal operations.
- **JMX connection handler.** This connection handler allows interactions with clients using the Java Management Extensions (JMX) framework and the Remote Method Invocation (RMI) protocol.
- **SNMP connection handler.** This connection handler is used to process SNMP requests to retrieve monitoring information described by MIB 2605. The supported SNMP protocols are SNMP V1, V2c, and V3.

17.1.5.1.2 Viewing All Connection Handlers

To display all configured connection handlers, along with their basic properties, use the `dsconfig list-connection-handlers` command.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-connection-handlers

Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
LDIF Connection Handler  : ldif : false   : -           : -
SNMP Connection Handler  : snmp : false   : 161         : -
```

17.1.5.2 Displaying the Properties of LDAP Connection Handler

To display the properties of LDAP connection handlers, use the `dsconfig` command.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-connection-handler-prop --handler-name "LDAP Connection Handler"
```

Depending on your configuration, the output will be similar to the following.

```
Property               : Value(s)
-----------------------:------------
allow-ldap-v2          : true
allow-start-tls        : false
allowed-client         : -
denied-client          : -
enabled                : true
keep-stats             : true
key-manager-provider   : -
listen-address         : 0.0.0.0
listen-port            : 1389
ssl-cert-nickname      : server-cert
ssl-cipher-suite       : -
ssl-client-auth-policy : optional
ssl-protocol           : -
trust-manager-provider : -
use-ssl                : false
```

17.1.5.3 Controlling Client LDAP Access to the Directory Server

You can specify a list of clients that may or may not access the directory server over LDAP. To do this, set the `allowed-client` or `denied-client` property of the LDAP connection handler. These properties take an IP address or subnetwork with subnetwork mask as values.

By default, these properties are not set and all clients are allowed access. Changes to these properties take effect immediately but do not interfere with connections that are already established.

This example permits access only to clients in the subnet mask `255.255.255.10`.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set allowed-client:255.255.255.10
```

17.1.5.4 Configuring the LDIF Connection Handler

The LDIF connection handler is disabled by default. This connection handler can be used to process changes in the server using internal operations. The changes to be processed are read from an LDIF file.

This section contains the following topics:

- Viewing Properties of the LDIF Connection Handler
- Enabling the JMX Alert Handler Through the LDIF Connection Handler

17.1.5.4.1 Viewing Properties of the LDIF Connection Handler

To view the default properties of the LDIF connection handler, you must use the `dsconfig` command.

For example, run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-connection-handler-prop --handler-name "LDIF Connection Handler"
```

Depending on your installation, the output will be similar to the following.

```
Property       : Value(s)
---------------:-------------------------
allowed-client : -
denied-client  : -
enabled        : false
ldif-directory : config/auto-process-ldif
poll-interval  : 5 s
```

The `ldif-directory` property specifies the directory in which the LDIF files are located. The connection handler checks if there are any files in this directory, at an interval specified by the `poll-interval` property. The connection handler then processes the changes contained in those files as internal operations and writes the result to an output file with comments indicating the result of the processing.

17.1.5.4.2 Enabling the JMX Alert Handler Through the LDIF Connection Handler

The example in this section demonstrates how to enable the JMX alert handler through the LDIF connection handler.

Perform the following steps:

1. Check the status of the JMX alert handler (disabled by default).

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     get-alert-handler-prop --handler-name "JMX Alert Handler"
   ```

   Depending on your installation, the output will be similar to the following.

   ```
   Property            : Value(s)
   --------------------:---------
   disabled-alert-type : -
   enabled             : false
   enabled-alert-type  : -
   ```

2. Create an LDIF file in the default LDIF directory that enables the JMX alert handler.

   ```
   $ cd ../config/
   $ mkdir auto-process-ldif
   $ cd auto-process-ldif/
   $ cat > disable-jmx.ldif << EOM
   > dn: cn=JMX Alert Handler,cn=Alert Handlers,cn=config
   > changetype: modify
   > replace: ds-cfg-enabled
   > ds-cfg-enabled: true
   > EOM
   $
   ```

3. After a period of time longer than `poll-interval`, recheck the status of the JMX alert handler.

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     get-alert-handler-prop --handler-name "JMX Alert Handler"

   Property            : Value(s)
   --------------------:---------
   disabled-alert-type : -
   enabled             : true
   enabled-alert-type  : -
   ```

17.1.5.5 Configuring the JMX Connection Handler

The JMX Connection Handler is used to interact with clients using the Java Management Extensions (JMX) protocol.

This section contains the following topics:

- Viewing Properties of the JMX Connection Handler
- Changing the Port on Which the Server Listens for JMX Connections

17.1.5.5.1 Viewing Properties of the JMX Connection Handler

To view the default properties of the JMX connection handler, you must use the `dsconfig` command.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-connection-handler-prop --handler-name "JMX Connection Handler"
```

Depending on your installation, the output will be similar to the following.

```
Property             : Value(s)
---------------------:------------
allowed-client       : -
denied-client        : -
enabled              : false
key-manager-provider : -
listen-port          : 1689
ssl-cert-nickname    : server-cert
use-ssl              : false
```

17.1.5.5.2 Changing the Port on Which the Server Listens for JMX Connections

The example in this section describes how to change the port on which the server listens for JMX connections to `1789`.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-connection-handler-prop \
  --handler-name "JMX Connection Handler" --set listen-port:1789
```

17.1.6 Configuring Network Groups Using `dsconfig`

You can configure network groups using `dsconfig` command. Network groups are the single entry point of all client requests to the Oracle Unified Directory. The network group handles all client interactions, dispatching them and delegating the treatment of the request to workflows. A client connection is associated to the network group with the highest priority and for which all the criteria are met.

During installation, a default network group with a priority of 1 is created. To set request filtering policies or resource limits, you must create a network group quality of service policy.

Each network group is associated with one or more workflows. The workflows provide access to a naming context (or suffix). By associating a workflow with a network group, you indicate to the network group which naming contexts are available. Typically to create a network group,

you would already have a workflow created. For information about workflows, see Configuring Workflows Using `dsconfig`.

All the commands in the following procedures specify the hostname (`-h`), the admin port (`-p`), the bind DN (`-D`), and the bind password file (`-j`). The examples use the `-X` option to trust all certificates.

This section describes how to configure network groups using the `dsconfig` command, and contains the following topics:

- About Network Group Creation
- Creating Network Groups
- Modifying Network Group Properties
- Creating a Network Group Quality of Service Policy
- Modifying a Network Group Quality of Service Policy
- Relocating the Root DSE Entry for a Network Group
- Customizing the Root DSE Entry for a Network Group

17.1.6.1 About Network Group Creation

You can create many network groups, in which case client requests will be handled by the network group with the highest priority, for which the criteria are met. Therefore, when you create a network group, you must consider all the network groups you plan to create, and the priority of each. The priority can be 0 or above, where 0 is the highest priority.

It is possible to create two network groups with the same priority. However, if two or more network groups have the same priority and match the client request, the network group that will handle the request is random, among those matching the client request. You should therefore specify a different priority for each network group.

The default properties of a new network group are as follows.

```
Property                Value(s)
-------------------------------------------------------------------------
allowed-auth-method     All authorization methods are allowed.
allowed-bind-dn         All bind DNs are allowed.
allowed-bind-id         All bind IDs are allowed.
allowed-client          All clients with addresses that do not match
                        an address on the deny list are allowed. If
                        there is no deny list, then all clients are
                        allowed.
allowed-portAll         All port numbers are allowed.
allowed-protocol        All supported protocols are allowed.
certificate-mapper      The global certificate mapper will be used.
denied-client           If an allow list is specified, then only
                        clients with addresses on the allow list are
                        allowed. Otherwise, all clients are allowed.
enabled                 true
generic-identity-mapper The global generic identity mapper will be
                        used.
gssapi-identity-mapper  The global GSSAPI identity mapper will be
                        used.
is-security-mandatory   false
priority                1
workflow                userroot0
```

17.1.6.2 Creating Network Groups

To create a network group, use the subcommand `create-network-group` option of the `dsconfig` command.

For example, run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group --group-name network-group1 --set enabled:true\
  --set workflow:workflow1 --set priority:1
```

After you have created a network group, you can associate a network group quality of service policy to it. For information about creating a quality of service policy, see Creating a Network Group Quality of Service Policy.

17.1.6.3 Modifying Network Group Properties

The network group properties filter the traffic and indicate how a request is directed.

This section contains the following topics:

- Understanding Network Group Properties
- Configuring Network Group Properties
- Setting an Allowed or Denied Client List

17.1.6.3.1 Understanding Network Group Properties

You can configure the network group properties to set the following criteria:

- the authentication method allowed between the client and the network group (`allowed-auth-method`).
- the bind DN allowed to connect to the network group (`allowed-bind-dn`).
- the list of clients authorized to access the Oracle Unified Directory (`allowed-client`), expressed by the IP address or name of the client. If no allowed client list is provided, all clients are allowed, assuming they are not listed in the denied client list.
- the protocol allowed to connect to the Oracle Unified Directory (`allowed-protocol`). If none is specified, then all protocols are allowed.
- the allowed port (s) to configure client connection to connect to the Oracle Unified Directory (`allowed-portAll`). If none is specified, then all the connection handlers ports are allowed.
- the name of the certificate mapper that should be used to match client certificates to user entries (`certificate-mapper`). If none is specified, then global certificate mapper is used.
- the list of clients not authorized to access the Oracle Unified Directory (`denied-client`). If no denied client list is provided, then all clients are authorized, assuming there is no limitation set by an allowed client list.
- the set of identity mappers that will be used by the network group to map an identity while performing SIMPLE, non-GSSAPI SASL bind requests and proxy authorization controls (`generic-identity-mapper`).
- the set of identity mappers that will be used by the network group to map an identity while performing GSSAPI/SASL bind requests (`gssapi-identity-mapper`).

- whether security between the client and the Oracle Unified Directory is always required (`is-security-mandatory`).
- the priority of the network group (`priority`). A client connection is first compared against the network group with the highest priority. If the client connection does not match its connection criteria, the client connection is compared against the network group with the next highest priority, and so on. If no network group is selected, the client connection is rejected.

17.1.6.3.2 Configuring Network Group Properties

You can modify network group properties, using the `dsconfig set-network-group-prop` command.

For example, to modify the *priority* of the network group, run the dsconfig command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-network-group-prop --group-name network-group1 --set priority:3
```

For example, you can ensure that no connections are accepted from the IP address `208.77.188.166`, by `network-group1` as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-network-group-prop --group-name network-group1 \
  --set denied-client:208.77.188.166
```

17.1.6.3.3 Setting an Allowed or Denied Client List

For `allowed-client` and `denied-client` lists, you must be aware of the name service configuration on the server. For example, if the name service knows the host as `myclienthost.example.com`, you must specify `myclienthost.example.com` as the value, and not just `myclienthost`. Similarly, if the name service knows the host as `myclienthost`, you must specify the value as `myclienthost`. If you do not know how the name service is configured, you should specify both the fully qualified domain name (for example `myclienthost.oracle.com`) and the short name (`myclienthost`) of the machine. Specifying multiple values will ensure that the name is resolved correctly. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-network-group-prop \
  --group-name network-group1 \
  --add denied-client:myhost \
  --add denied-client:myhost.example \
  --add denied-client:myhost.example.com
```

To avoid any issues, use the IP address for clarity.

If you use `localhost` or the name of the local machine when connecting to Oracle Unified Directory, the IP addresses of the client will be different. To prevent connections from the localhost, specify both `localhost` and the name of the local machine in the list of denied clients.

17.1.6.4 Creating a Network Group Quality of Service Policy

You can, optionally, associate a quality of service (QoS) policy with a network group. A QoS policy applies additional filtering criteria to client connections to determine how the network group handles the request.

Oracle Unified Directory supports five types of QoS policy:

- Request filtering policy
- Resource limits
- Affinity
- Referral
- Subtree Access Control QoS Policy

> **Note:**
>
> OUDSM accesses an Oracle Unified Directory instance over the administration connector. The administration connector is not subject to the QoS policies defined for a network group. OUDSM therefore bypasses the QoS policies defined for a network group. For more information, see Managing Administration Traffic to the Server.

To create a network group quality of service policy, use the `dsconfig create-network-group-qos-policy` command. You must specify the name of the network group to which the quality of service policy applies, and the type of quality of service policy.

This section contains the following topics:

- Creating a Request Filtering Quality of Service Policy
- Creating a Resource Limit Quality of Service Policy
- Creating an Affinity Quality of Service Policy
- Creating a Referral Quality of Service Policy
- Creating a Subtree Access Control Quality of Service Policy

17.1.6.4.1 Creating a Request Filtering Quality of Service Policy

A request filtering policy applies the following criteria to an incoming client request:

- `allowed-attributes`: list of attributes that can be specified in the filter of a search request.
- `allowed-operations`: type of operation accepted by the network group. For example, you can specify that a network group should accept only read requests.
- `allowed-search-scopes`: scope of a search accepted, for example one-level only.
- `allowed-subtrees`: list of specific subtrees that can be specified as a base DN in a search request.
- `prohibited-attributes`: list of attributes which, if specified in the filter of a search request, will be rejected.
- `prohibited-subtrees`: list of specific subtrees, which if specified as base DNs in a search request, will be rejected.

The following example defines a request filtering policy that ensures that users can only search and not modify data:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group-qos-policy --group-name network-group1 \
  --type request-filtering --set allowed-operations:search
```

17.1.6.4.2 Creating a Resource Limit Quality of Service Policy

A resource limit policy sets specific limits on the client connections that can access the server through that network group. The following limits can be defined:

- `max-concurrent-ops-per-connection`: Specifies the maximum number of simultaneous operations per established connection. To run the server in synchronous mode, set the maximum to 1.
- `max-ops-per-connection`: Specifies the maximum number of operations per connection.
- `max-connections`: Specifies the maximum number of concurrent client connections to the server. If you do not set a maximum number of connections, the server limit is used.
- `max-connections-from-same-ip`: Specifies the maximum number of connections from the same IP address. Set this parameter if you want to avoid Denial of Service attacks. This parameter should not be set if you know that most requests typically come from the same client.
- `max-ops-per-interval`: Specifies the maximum number of operations per specified interval. For example, a setting of 1,000 will limit the number of operations to 1,000 per the interval set using `max-ops-interval`.
- `max-ops-interval`: Specifies the interval during which the number of operations is counted for the `max-ops-per-interval` parameter. For example, an interval set to one second results in operations being counted per second. The limit (`max-ops-per-interval`) is checked and enforced during each interval.
- `min-substring-length`: Specifies the minimum search string length. The shorter the search string, the more results that need to be found and displayed. It is therefore useful to set a minimum search string length in the substring search filter to limit the resources that are used.
- `size-limit`: Specifies the maximum number of entries that can be returned to the client during a single search operation. It is recommended that you keep the default setting for this property.
- `time-limit`: Specifies the maximum amount of time that a search operation should take. It is recommended that you keep the default setting for this property.
- `idle-time-limit`: Specifies the maximum duration a client connection can remain active after its previous completed operation. A value of `0` second implies that there is no idle time limit.

The following example defines a resource limit policy that ensures that a user enters a search string of at least five characters, to limit the number of return values:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group-qos-policy --group-name network-group1 \
  --type resource-limits --set min-substring-length:5
```

17.1.6.4.3 Creating an Affinity Quality of Service Policy

In a load balancing deployment, you can use *affinity* to override the regular routing process. The properties of the affinity policy determine the routing process that should be followed.

You can configure the following properties:

- `affinity-policy`: Specifies the routing policy to use.

The affinity policy can take one of the following values:

- `all-requests-after-first-request`
- `all-requests-after-first-write-request`
- `all-write-requests-after-first-write-request`
- `first-read-request-after-write-request`

Specific operations will set affinity, depending on the affinity policy. For the first policy in the previous list (`all-requests-after-first-request`) all operations will set affinity. For the remaining policies (`all-requests-after-first-write-request`, `all-write-requests-after-first-write-request`, and `first-read-request-after-write-request`) only an `ADD`, `DELETE`, `MOD` or `MODDN` operation will set affinity.

- `affinity-timeout`: Defines the duration during which the affinity applies.

Even when affinity has been set by a previous operation, the load balancing algorithm is only bypassed in specific situations, depending on the affinity policy and the current operation type. If the affinity policy is `all-requests-after-first-request` or `all-requests-after-first-write-request`, the affinity route will be used for every operation type, unless the affinity timeout has expired. If the affinity policy is `all-write-requests-after-first-write`, the affinity route will be used for any `ADD`, `DELETE`, `MOD` or `MODDN` operation, unless the timeout has expired. The affinity route will not be used for other operations. If the affinity policy is `first-read-request-after-write-request`, the affinity route will be used for all operations *except* `ADD`, `DELETE`, `MOD` or `MODDN` operations, unless the timeout has expired.

The following example sets an affinity policy that can be set by any operation and used for all operations, for a maximum of sixty seconds.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group-qos-policy --group-name network-group1 \
  --type affinity --set affinity-timeout:60s \
  --set affinity-policy:all-requests-after-first-request
```

> **Note:**
>
> The affinity feature can be used with all load balancing algorithms except for the failover algorithm. With the failover algorithm, only one route is active at a time. The active route changes when the remote server goes down, so all connections to the remote server are broken. Affinity can therefore not apply in a failover scenario.

17.1.6.4.4 Creating a Referral Quality of Service Policy

You can configure the behavior of a proxy server when a referral is received from the remote LDAP server by defining a referral quality of service policy. The referral itself must be defined on the remote LDAP server.

When you create a network group quality of service, you can set the following referral properties:

- Maximum number of hops supported (`referral-hop-limit`) when the referral policy is set to `follow`. The default is set to 5.
- Define the type of referral policy (`referral-policy`), such as `discard`, `forward`, or `follow`. This property defines how a referral will be treated by the network group.

For example, the referral-policy is set by default to `forward`. You can change it to `discard` or to `follow`, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group-qos-policy --group-name network-group1 \
  --type referral --set referral-policy:follow
```

17.1.6.4.5 Creating a Subtree Access Control Quality of Service Policy

A subtree access control policy applies the following criteria to an incoming client request:

- `allowed-attributes`: list of attributes that can be specified in the filter of a search request.
- `allowed-bind-dn`: list of allowed bind DN.
- `allowed-operations`: type of operations accepted by the network group. For example, you can specify that a network group should accept only read requests.
- `allowed-search-scopes`: scope of a search accepted, for example one-level only.
- `allowed-subtrees`: list of specific subtrees that can be specified as a base DN in a search request.
- `prohibited-attributes`: list of attributes which, if specified in the filter of a search request, will be rejected.
- `prohibited-bind-dn`: list of prohibited bind DN.
- `prohibited-subtrees`: list of specific subtrees, which if specified as base DNs in a search request, will be rejected.

The following example defines a subtree access control policy that ensures only configured bind DN will have access to subtree `dc=example,dc=com`:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-network-group-qos-policy-advanced --group-name network-group1 \
  --type subtree-access-control --set base-dn:dc=example,dc=com \
  --set allowed-bind-dn:uid=testuser,ou=People,dc=example,dc=com \
  --advanced-name subtreeaccesspolicy
```

17.1.6.5 Modifying a Network Group Quality of Service Policy

To modify a QoS policy, use the `dsconfig set-network-group-qos-policy-prop` command, specifying the network group name and the policy type.

The following example sets the minimum search string limit of a resource limits quality of service policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-network-group-qos-policy-prop --group-name network-group1 \
  --policy-type resource-limits --set min-substring-length:5
```

17.1.6.6 Relocating the Root DSE Entry for a Network Group

The Root DSE is a special entry that provides information about the server's name, version, naming contexts, and supported features. The Root DSE entry of a network group can be in a local server or a remote server.

To relocate the Root DSE, use the `dsconfig set-network-group-prop` command, as shown in the following example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-network-group-prop --group-name network-group1 \
  --set relocated-rootdse-workflow-element:<new rootDSE workflow element>  \
```

The value of the `relocated-rootdse-workflow-element` property is the workflow element where a Root DSE can be found (This is the entry returned by a search on the null DN).

17.1.6.7 Customizing the Root DSE Entry for a Network Group

The default Root DSE view may not display all the information you want to view. For example, by default the Root DSE view may not display all `supportedControls` you want to see. You can customize the Root DSE view.

To customize the Root DSE view:

1. Generate a Root DSE LDIF template. For example:

   ```
   ldapsearch -b "" -s base "(objectclass=*)" "*" + > rootDse.ldif
   ```

2. Customize the LDIF.

   For example, you can add or remove `supportedControls`.

3. Create an LDIF back end specifying a single space as DN. For example:

   ```
   dsconfig create-workflow-element -p $APORT -n -X -D "$ADN" -j $APWF --type ldif-
   local-backend
   --element-name customRootDSE
   --set ldif-file:$PWD/rootDse.ldif
   --set is-private-backend:true
   --set writability-mode:disabled
   --set base-dn:" "
   --set enabled:true
   ```

4. Redirect the Root DSE toward the LDIF back end. For example:

   ```
   dsconfig set-network-group-prop -p $APORT -n -X -D "$ADN" -j $APWF
   --group-name network-group
   --set relocated-rootdse-workflow-element:customRootDSE
   ```

5. Restart the server.

17.1.7 Configuring Workflows Using `dsconfig`

You can configure workflows using the `dsconfig` command. A workflow is the link between the network group and the naming context (suffixes). It defines the naming context that will be accessible for a given network group, when handling a request to a load balancing or distribution configuration. To create a workflow, you must already have a load balancing or distribution workflow element created.

For information on workflow elements, see Configuring Workflow Elements Using `dsconfig`.

The topics described in this section contain examples to configure workflows using the `dsconfig` command. All the commands in the following procedures specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN (`-D`), and the bind password file (`-j`). The examples use the `-X` option to trust all certificates.

This section contains the following topics:

- Viewing Workflow Properties
- Creating a Workflow

17.1.7.1 Understanding Privacy Settings of the Remote LDAP Servers

The proxy automatically creates several private workflows. *Do not* modify or delete these workflows. When configuring workflows, you must consider the privacy settings of the remote LDAP servers. Table 17-1 describes these privacy settings.

Table 17-1 Remote LDAP Server Privacy Settings

Privacy Setting	Description
LDIFBackend	Privacy is defined by the `ds-cfg-is-private-backend` property. The default setting for this flag is `private`, but you can change it.
JEB backend	Always public, and contains user data.
Config File Handler backend	Always private.
Backup backend	Always private.
Schema backend	Always private.
Tasks backend	Always private.
Monitor backend	Always private.
Truststore backend	Always private.

17.1.7.2 Listing Existing Workflows

To display all the workflows configured on a server instance, use the `dsconfig list-workflows` command. The following example shows the default workflow configured on a proxy server instance:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-workflows

Workflow       : Type    : enabled
---------------:---------:--------
workflow1      : generic : true
```

17.1.7.3 Viewing Workflow Properties

To view the properties of a specific workflow, use the `dsconfig get-workflow-prop` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-workflow-prop --workflow-name workflow1

Property             : Value(s)
---------------------:------------------
base-dn              : "ou=people,o=test"
enabled              : true
workflow-element     : load-bal-we1
access-control-group : Local Backends
virtual-aci-mode     : false
```

The `base-dn` indicates the base DN used for the workflow, and therefore for the deployment using that workflow. The `workflow-element` property indicates the workflow element that will process the requests.

> **Note:**
>
> The `base-dn` property is read-only and cannot be modified.

17.1.7.4 Creating a Workflow

Each workflow is associated with a workflow element. When you create a workflow, you must specify the associated workflow element name (`--set workflow-element`). In other words, you must create the workflow element before attaching it with a workflow. See Configuring Workflow Elements Using `dsconfig`.

Each workflow is associated with an access control group. When you create a workflow, you can specify the associated access control group name (`--set access-control-group`). By default, the `Local Backends` access control group is used. If you want to specify a specific access control group, then you must already have created the access control group. For more information about configuring access control groups, see Configuring Access Control Groups With `dsconfig`.

You can enable virtual ACIs for each workflow. To enable the virtual ACIs feature, you can set the `virtual-aci-mode` parameter to `true`, using the command `--set virtual-aci-mode:true`.

To create a workflow, use the `dsconfig create-workflow` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-workflow \
  --workflow-name workflow1 \
  --set base-dn:ou=people,o=test \
  --set enabled:true \
  --set workflow-element:load-bal-we1
```

17.1.8 Configuring Workflow Elements Using `dsconfig`

Workflow elements are part of a routing structure, and they are linked to workflows. For a directory server instance, DB local workflow elements are associated with a physical database.

For more information about workflow elements, including available types and how they are used, see Understanding Workflow Elements.

A proxy deployment must include LDAP proxy workflow elements and either a load balancing or distribution workflow element.

The topics described in this section explain how to configure workflow elements using the `dsconfig` command. All the commands in the following procedures specify the hostname (`-h`), the administration port (`-p`), the bind DN (`-D`), and the bind password file (`-j`). The examples use the `-X` option to trust all certificates.

This section contains the following topics:

- Listing Workflow Elements
- Creating Workflow Elements

- Modifying Workflow Elements

17.1.8.1 Listing Workflow Elements

To display all the configured workflow elements, use the `dsconfig list-workflow-elements` command.

The following example shows the default workflow elements for a directory server instance.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-workflow-elements

Workflow Element : Type               : enabled
-----------------:-------------------:--------
adminRoot        : ldif-local-backend : true
userRoot         : db-local-backend   : true
virtualAcis      : db-local-backend   : true
```

The following example shows the default workflow elements for a proxy server instance, deployed for load balancing:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-workflow-elements

Workflow Element : Type               : enabled
-----------------:-------------------:--------
adminRoot        : ldif-local-backend : true
load-bal-we1     : load-balancing     : true
proxy-we1        : proxy-ldap         : true
proxy-we2        : proxy-ldap         : true
```

17.1.8.2 Creating Workflow Elements

To create workflow elements in interactive mode, use the `dsconfig create-workflow-element` command. If you configured a proxy instance during the setup, the required workflow elements will already have been created.

You can create the following types of workflow elements:

- DB Local Backend. For more information, see Creating a DB Local Backend Workflow Element.
- Load balancing. For more information, see Creating a Load Balancing Workflow Element.
- Distribution. For more information, see Creating a Distribution Workflow Element.
- Proxy LDAP. For more information, see Creating the Proxy LDAP Workflow Elements.
- DN renaming. For more information, see Performing DN Renaming.
- Kerberos Authentication. For more information, see Creating a Kerberos Workflow Element Using `dsconfig`.

17.1.8.2.1 Creating a DB Local Backend Workflow Element

A Local Backend workflow element provides access to a back end in a directory server instance. To create a new Local Backend workflow element, use the `dsconfig create-workflow-element` command, specifying one or more base DNs that will be accessed through the workflow element.

A single back end can be responsible for one or more base DNs. No two back ends may have the same base DN, but one back end can have a base DN that is below a base DN provided by another back end. If any of the base DNs is subordinate to a base DN for another back end, then all base DNs for that back end must be subordinate to that same base DN.

The following example creates and enables a Local Backend workflow element to access the base DN `ou=admins,dc=example,dc=com`.

```
$ dsconfig create-workflow-element -h localhost -p 4444 -D "cn=directory manager"\
  -j pwd-file -X -n --element-name admins --type db-local-backend \
  --set base-dn:ou=admins,dc=example,dc=com --set enabled:true
```

17.1.8.3 Modifying Workflow Elements

Once you have created a workflow element, you can modify its properties using the `dsconfig set-workflow-element-prop` command.

17.1.9 Configuring Plug-Ins Using `dsconfig`

Plug-ins are responsible for providing custom logic in the course of processing an operation or at other well-defined points within the directory server. The `dsconfig` command is used to manage the configuration of the directory server.

For information about using `dsconfig`, see Managing the Server Configuration Using `dsconfig`.

This section covers the following topics:

- Understanding the Plug-In Types
- Modifying the Plug-In Configuration

17.1.9.1 Understanding the Plug-In Types

The `dsconfig plugin-type` property can be used to configure a plug-in to use one or more of the numerous plug-in types supported by the server. You cannot add a new default plug-in type to the configuration of an existing plug-in. Although, you can remove one or more of the default plug-in type values from a plug-in's configuration, you must take care when doing this. Usually a plug-in has been engineered to support its default plug-in types for a reason. Removing one or more plug-in types might endanger the safe operation of the directory server.

Most of the plug-ins support more than one type, and multiple plug-ins are sometimes defined with the same plug-in type. The order in which these plug-ins are invoked during processing is undefined. If a specific order is required (for example, if the processing performed by one plug-in depends on the result of another), you can specify the order in which the plug-ins are invoked. For more information, see Configuring Plug-In Invocation Order.

17.1.9.2 Modifying the Plug-In Configuration

The following sections show various examples of managing plug-in configuration using `dsconfig`. `dsconfig` uses the administration connector to access the server. All of the examples in this section assume that the administration connector is listening on the default port (4444) and that the command is accessing the server running on the local host. If this is not the case, the `--port` and `--hostname` options must be specified.

The dsconfig command always accesses the server over a secured connection with certificate authentication. If you run `dsconfig` in interactive mode, you are prompted about how you want

to trust the certificate. If you run `dsconfig` in non-interactive mode (that is, with the `-n` option) you must specify the `-X` or `--trustAll` option, otherwise the command will fail.

This section describes examples to manage plug-in configuration, and covers the following topics:

- Displaying a List of Plug-Ins
- Creating a New Plug-In
- Enabling or Disabling a Plug-In
- Displaying and Configuring Plug-In Properties
- Configuring Plug-In Invocation Order

17.1.9.2.1 Displaying a List of Plug-Ins

The example in this section shows a directory server configured with the current supported plug-ins. For a description of these plug-ins and their purpose, see "The Plug-In Configuration" in the *Configuration Reference for Oracle Unified Directory*.

Use `dsconfig` to display the list of plug-ins that are currently configured.

```
$ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
  list-plugins
```

Depending on your installation, the output will be similar to the following.

```
Plugin                          : Type                            : enabled
--------------------------------:---------------------------------:--------
7-Bit Clean                     : seven-bit-clean                 : false
Change Number Control           : change-number-control           : true
Entry UUID                      : entry-uuid                      : true
LastMod                         : last-mod                        : true
LDAP Attribute Description List : ldap-attribute-description-list : true
Password Policy Import          : password-policy-import          : true
Profiler                        : profiler                        : true
Referential Integrity           : referential-integrity           : false
Replication LDIF Import         : replication-ldif-import         : true
UID Unique Attribute            : unique-attribute                : false
```

The output of the command shows (from left to right):

- **Plug-in.** The name of the plug-in, usually descriptive of what it does.
- **Type.** The type of plug-in. It is possible to have more than one plug-in of a specific type.
- **Enabled.** Plug-ins can be enabled or disabled. Disabled plug-ins remain in the server configuration but do not perform any processing.

17.1.9.2.2 Creating a New Plug-In

The easiest way to configure plug-ins is to use `dsconfig` in interactive mode. Interactive mode walks you through the plug-in configuration, and is therefore not documented here.

The following example shows how to create and enable a new Password Policy Import Plug-in using `dsconfig` in non-interactive mode.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-plugin --type password-policy-import \
  --plugin-name "My Password Policy Import Plugin" --set enabled:true
```

17.1.9.2.3 Enabling or Disabling a Plug-In

You can enable or disable a plug-in by setting the `enabled` property to `true` or `false`. This example disables the Password Policy Import plug-in created in the previous example.

Run the `dsconfig` command to disable the new Password Policy Import plug-in.

```
$ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
  set-plugin-prop --plugin-name "My Password Policy Import Plugin" \
  --set enabled:false
```

17.1.9.2.4 Displaying and Configuring Plug-In Properties

To display the properties of a plug-in, use the `get-plugin-prop` subcommand. To change the properties of a plug-in, use the `set-plugin-prop` subcommand. This example displays the properties of the plug-in created in the previous example, then enables the plug-in and sets the default authentication password storage scheme to Salted SHA-512.

1. Display the plug-in properties.

   ```
   $ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
     get-plugin-prop --plugin-name "My Password Policy Import Plugin"
   ```

   Depending on your installation, the output will be similar to the following.

   ```
   Property                            : Value(s)
   ------------------------------------:---------
   default-auth-password-storage-scheme : -
   default-user-password-storage-scheme : -
   enabled                              : false
   ```

2. Enable the plug-in and set the default authentication password storage scheme to Salted SHA-512.

   ```
   $ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
     set-plugin-prop --plugin-name "My Password Policy Import Plugin" \
     --set enabled:true\
     --set default-auth-password-storage-scheme:"Salted SHA-512"
   ```

3. Display the plug-in properties again to verify the change.

   ```
   $ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
     get-plugin-prop --plugin-name "My Password Policy Import Plugin"

   Property                            : Value(s)
   ------------------------------------:---------------
   default-auth-password-storage-scheme : Salted SHA-512
   default-user-password-storage-scheme : -
   enabled                              : true
   ```

17.1.9.2.5 Configuring Plug-In Invocation Order

By default, the order in which plug-ins are invoked is undefined. You can use the `set-plugin-root-prop --set plugin-type:value` subcommand to specify that plug-ins be invoked in a specific order. The *value* in this case is the plug-in order, expressed as a comma-delimited list of plug-in names. The plug-in order string should also include a single asterisk element, which is a wildcard that will match any plug-in that is not explicitly named.

This example specifies that the Entry UUID plug-in should be invoked before any other pre-operation add plug-ins.

1. Display the current plug-in invocation order.

```
$ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
  get-plugin-root-prop

Property                                  : Value(s)
------------------------------------------:---------
plugin-order-intermediate-response        : -
plugin-order-ldif-export                  : -
plugin-order-ldif-import                  : -
plugin-order-post-connect                 : -
...
```

2. Set the plug-in order.

```
$ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j pwd-file -X -n \
  set-plugin-root-prop --set plugin-order-pre-operation-add:"Entry UUID,*"
```

> **Note:**
>
> Plug-in order values are not validated. Values that do not match defined plug-ins are ignored.

17.1.10 Configuring Suffixes with `dsconfig`

Oracle Unified Directory allows you to configure multiple suffixes, either during the setup or later.

This section contains the following topics:

- Configuring Suffixes with `dsconfig` During Setup
- Configuring Suffixes with `dsconfig` on a Running Server

You can also use `dsconfig` in interactive mode to achieve the configuration described in the following sections.

17.1.10.1 Configuring Suffixes with `dsconfig` During Setup

You can configure suffixes with the `dsconfig` command during the setup by creating the base entries.

You can use any one method listed here to create the base entries, for example, `dc=example,dc=com;dc=other,dc=com;dc=test,dc=com.`

- Create the base entries using the following command:

```
oud-setup --cli --baseDN dc=example,dc=com --baseDN dc=test,dc=com --baseDN \
dc=other,dc=com --addBaseEntry --ldapPort 2389 --adminConnectorPort 24444 \
--rootUserDN cn=Directory Manager --rootUserPassword password --no-prompt \
--noPropertiesFile
```

- Create the base entries with sample data using the following command:

```
oud-setup --cli --baseDN dc=example,dc=com --baseDN dc=test,dc=com --baseDN \
dc=other,dc=com --sampleData 15 --ldapPort 2389 --adminConnectorPort 24444 \
```

```
--rootUserDN cn=Directory Manager --rootUserPassword password --no-prompt \
--noPropertiesFile
```

You can now access data below all the suffixes without additional configuration.

17.1.10.2 Configuring Suffixes with `dsconfig` on a Running Server

You can configure suffixes on a running server instance using the `dsconfig` command or using OUDSM. For more information about configuring suffixes with OUDSM, see Creating a Suffix.

To configure suffixes with the `dsconfig` command, perform the following steps:

1. Add the base DN to your Local Backend workflow element.

```
dsconfig set-workflow-element-prop \
--element-name userRoot \
--add base-dn:dc=example2,dc=com \
--hostname localhost \
--port 24444 \
--trustAll \
--bindDN cn=directory manager \
--bindPassword ****** \
--no-prompt
```

2. Create a workflow for your new base DN.

```
dsconfig create-workflow \
--set base-dn:dc=example2,dc=com \
--set enabled:true \
--set workflow-element:userRoot \
--type generic \
--workflow-name dc=example2,dc=com \
--hostname localhost \
--port 24444 \
--trustAll \
--bindDN cn=directory manager \
--bindPassword ****** \
--no-prompt
```

3. Add your new workflow to your network group.

```
dsconfig set-network-group-prop \
--group-name network-group \
--add workflow:dc=example2,dc=com \
--hostname localhost \
--port 24444 \
--trustAll \
--bindDN cn=directory manager \
--bindPassword ****** \
--no-prompt
```

4. Create the base entry, `dc=example2,dc=com`.
5. Populate your new suffix with the required entries.

17.1.11 Configuring Access Control Groups With `dsconfig`

An access control group determines the ACIs that apply to specific operation. Each workflow is associated with an access control group which defines the list of ACIs that apply to operations handled by this workflow. By default, an access control group known as Local Backends exists. This access control group contains all ACIs coming from user data and cannot be deleted.

The section describes how to configure access control groups with the `dsconfig` command, and contains the following topics:

- Creating Access Control Groups
- Deleting Access Control Groups

17.1.11.1 Creating Access Control Groups

Run the `dsconfig` command to create an access control group as follows:

```
dsconfig create-access-control-group --group-name group1
```

17.1.11.2 Deleting Access Control Groups

Run the `dsconfig` command to delete an access control group as follows:

```
dsconfig delete-access-control-group-prop --group-name group1
```

> **Note:**
>
> You cannot delete `Local Backends` access control group. You can only delete those access control groups that are not associated with any workflow. Deleting an access control group will delete all ACIs contained in that access control group.

17.2 Managing Suffixes Using `manage-suffix`

The `manage-suffix` command allows you to create and manage local suffixes that store data in a local database. Although you can also use `dsconfig` to create and manage suffixes, the `manage-suffix` tool is a dedicated tool, and much easier to use.

For example, the `manage-suffix` command requires only a DN to be able to create a suffix. To compare the tools, see also Configuring Suffixes with `dsconfig`.

Use `manage-suffix` utility when you want to integrate Oracle Unified Directory with other Oracle components such as Enterprise User Security, Database Net Services, and E-Business Suite.

Before you can add data to an Oracle Unified Directory server, you must define the suffix or suffixes that will contain the data.

The following examples illustrate how to use the `manage-suffix` command:

- Creating an Integrated Suffix Using `manage-suffix`
- Creating a Non-Integrated Suffix Using `manage-suffix`
- Viewing Suffix Information
- Modifying a Suffix Configuration
- Deleting a Suffix Using `manage-suffix`

17.2.1 Creating an Integrated Suffix Using `manage-suffix`

When you create an integrated suffix using `manage-suffix`, the tool prepares Oracle Unified Directory for integration with other Oracle components.

If a local database workflow element already exists, the suffix is created and configured in the existing local database workflow element. If no user suffix existed in the server before running the utility, then the user suffix is created and configured in a new local database workflow element. If no network group is specified, and only the default network group is defined, the suffix is registered in the default network group. If no network group is defined, a new network group is created and the suffix is registered in the new network group.

You can use the `manage-suffix` utility in non-interactive or interactive CLI mode. For more information, see:

- Creating a Suffix Using the Non-Interactive CLI Mode
- Creating a Suffix Using the Interactive CLI Mode

17.2.1.1 Creating a Suffix Using the Non-Interactive CLI Mode

The example in this section creates two suffixes, provisioned with base entry only, and configured for integration with Enterprise User Security (EUS) using the non-interactive CLI mode.

To create a suffix using the non-interactive CLI mode, run the `manage-suffix` command as follows:

```
$ manage-suffix create \
          --baseDN dc=suffix1 \
          --baseDN dc=suffix2 \
          --entries base-entry \
          --integration eus \
          --hostname host1.local \
          --port 4444 \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile ****** \
          --trustAll \
          --no-prompt
```

17.2.1.2 Creating a Suffix Using the Interactive CLI Mode

The example in this section creates two suffixes, provisioned with base entry only, and configured for integration with Enterprise User Security (EUS) using the interactive CLI mode.

To create a suffix using the non-interactive CLI mode, run the `manage-suffix` command as follows:

```
$ manage-suffix -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X

What do you want to do?

    1)  Create Suffixes
    2)  Delete Suffixes
    3)  Update Suffixes
    4)  List the Suffixes

    q)  quit
```

```
Enter choice [1]:

Reading Configuration ..... Done.

Provide the DNs of the suffixes to be created.  Leave empty when you have
finished.
DN: dc=suffix1
DN: dc=suffix2
DN:

Specify the Oracle components with which the suffixes can integrate.

    1)  No Integration
    2)  Generic: Database Net Services and EBS (E-Business Suite)
    3)  EUS (Enterprise User Security), Database Net Services and EBS
        (E-Business Suite)

    c)  cancel

Enter choice [1]: 3

Options to populate the suffix:

    1)  Only create the base entry
    2)  Load automatically-generated sample data

    c)  cancel

Enter choice [1]:

Creating suffixes ..... Done.

Adding Data ..... Done.

Updating Oracle Integration ...... Done.
```

17.2.2 Creating a Non-Integrated Suffix Using `manage-suffix`

You can create a non-integrated suffix using the `manage-suffix` command. In the following examples, a new suffix is created in different DB and using a different network group than in the previous examples. The new suffix is *not* configured for integration with an Oracle product.

This section contains the following topics:

- Creating a Non-Integrated Suffix Using the Non-Interactive CLI Mode
- Creating a Non-Integrated Suffix Using the Interactive CLI Mode

17.2.2.1 Creating a Non-Integrated Suffix Using the Non-Interactive CLI Mode

You can create a non-integrated suffix using the non-interactive CLI mode by running the `manage-suffix create` command with the following arguments:

```
$ manage-suffix create \
          --baseDN cn=nointegrated \
          --entries base-entry \
          --integration no-integration \
          --networkGroup network-group2 \
          --workflowElement userRoot2 \
          --dbPath config/db \
          --hostname host1.local \
```

```
--port 4444 \
--bindDN cn=Directory\ Manager \
--bindPasswordFile ****** \
--trustAll \
--no-prompt
```

17.2.2.2 Creating a Non-Integrated Suffix Using the Interactive CLI Mode

You can create a non-integrated suffix using the interactive CLI mode by running the `manage-suffix create` command with the following arguments:

```
$ manage-suffix -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X --advanced

Reading Configuration ..... Done.

What do you want to do?

    1)  Create Suffixes
    2)  Delete Suffixes
    3)  Update Suffixes
    4)  List the Suffixes

    q)  quit

Enter choice [1]:

Provide the DNs of the suffixes to be created.  Leave empty when you have
finished.
DN: cn=nointegrated
DN:

Choose the network groups (separated by commas) that must expose the suffixes.

    1)  network-group
    2)  Create a new network group

    c)  cancel

Enter one or more choices separated by commas [1]: 2

Network Group Name: network-group2
Choose the Local DB workflow element where you want to store data.

    1)  userRoot
    2)  Create a new Local DB workflow element

    c)  cancel

Enter choice [1]: 2

Local DB Name: userRoot2

Provide the path where the data will be stored.  It can be an absolute path or
a relative path to the server location.
DB Path: [db]: config/db

Specify the Oracle components with which the suffixes can integrate.

    1)  No Integration
    2)  Generic: Database Net Services and EBS (E-Business Suite)
    3)  EUS (Enterprise User Security), Database Net Services and EBS
        (E-Business Suite)
```

```
    c)  cancel

Enter choice [1]:

Options to populate the suffix:

    1)  Only create the base entry
    2)  Leave the database empty
    3)  Load automatically-generated sample data

    c)  cancel

Enter choice [1]:

Creating suffixes ..... Done.

Adding Data ..... Done.

Some new network groups have been created.  If the contents of the suffixes
are not exposed when performing LDAP operations, you must check the
configuration of the network groups and update them accordingly to your LDAP
clients.
```

In this example, a new suffix is created in a new local database workflow element (`userRoot2`), and in a new network group (`network-group2`). The `--advanced` option is required in this example because the administrator wants to create a new network group and a new local database workflow element for the new suffix.

17.2.3 Viewing Suffix Information

Use the `manage-suffix list` command to view information about local, configured suffixes. Use the `--advanced` option when you want to view information about internal suffixes with advanced configurations.

For example, use the `--advanced` option when you want to view internal suffixes used to configure integration among Oracle Unified Directory and other Oracle products.

You can run `manage-suffix list` in non-interactive or interactive CLI mode. For a complete list of options and usage, run the following command:

```
$ manage-suffix list --help
```

This section contains the following examples:

- Displaying Suffix Information Using Default Options
- Displaying a Set of Suffixes
- Displaying Internal Suffixes

17.2.3.1 Displaying Suffix Information Using Default Options

You can display the suffix information using default `manage-suffix` options.

Run the `manage-suffix` command as follows:

```
$ manage-suffix -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n
```

```
Base DN          : Entries : Oracle Integration [1] : Type            : Other Information
-----------------:---------:-------------------------:----------------:---------------------------
cn=nointegrated  : 1       : No Integration         : Local Database  : Not Replicated (Use
                 :         :                        :                 : 'dsreplication enable' to
                 :         :                        :                 : enable replication)
dc=suffix1       : 1       : EUS [2]                : Local Database  : Not Replicated (Use
                 :         :                        :                 : 'dsreplication enable' to
                 :         :                        :                 : enable replication)
dc=suffix2       : 1       : EUS [2]                : Local Database  : Not Replicated (Use
                 :         :                        :                 : 'dsreplication enable' to
                 :         :                        :                 : enable replication)

[1] You can update the integration using the 'manage-suffix update' sub-command.
[2] The suffix can integrate its contents with EUS (Enterprise User Security), EBS (E-Business Suite)
    and Database Net Services.
```

17.2.3.2 Displaying a Set of Suffixes

To display only a set of suffixes, use the `--baseDN` argument to specify which suffixes must be displayed. If no `--baseDN` argument is provided, all suffixes are displayed. You can also use the `--advanced` argument to display the internal suffixes.

The `--listDataToDisplay` argument is an informative argument that lists and describes the different allowed values for the argument `--dataToDisplay`.

Use the `--dataToDisplay` argument to specify which information is displayed.

The example in this section provides information for only `suffix dc=suffix2` and only the network group and workflow element are displayed.

Run the `manage-suffix` command as follows:

```
$ manage-suffix list --baseDN dc=suffix2 -X --dataToDisplay network-group --
dataToDisplay workflow-element -j pwd-file -X -n

Reading Configuration ..... Done.

Base DN     : Wfe [1]   : N.G. [2]
------------:-----------:---------------
dc=suffix2  : userRoot  : network-group

[1] The name of the configuration entity (workflow element) containing the data.
If the data of the data is not stored locally, it returns the name of the first
workflow element associated with the suffix
[2] The name of the network groups that expose the contents of this suffix
```

17.2.3.3 Displaying Internal Suffixes

You can display internal suffixes using the `--advanced` option of the `manage-suffix` command.

Run the `manage-suffix` command as follows:

```
$ manage-suffix -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n --
advanced

Reading Configuration ..... Done.
```

```
Base DN                       : Entries : Oracle Integration [1]         : Type             : Other Information
------------------------------:---------:--------------------------------:------------------:----------------------------
cn=admin data                 : 7       : --                             : Local LDIF File  : Replication Administration
                              :         :                                :                  : Data
cn=nointegrated               : 1       : No Integration                 : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=OracleContext              : 26      : Integration Infrastructure [2] : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=OracleSchemaVersion        : 3       : Integration Infrastructure [2] : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=subschemasubentry          : 1       : Integration Infrastructure [2] : Renaming Schema  : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=virtual acis               : 0       : --                             : Local Database   : Virtual ACIs
dc=suffix1                    : 1       : EUS [3]                        : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=OracleContext,dc=suffix1   : 17      : Integration Infrastructure [2] : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
dc=suffix2                    : 1       : EUS [3]                        : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)
cn=OracleContext,dc=suffix2   : 17      : Integration Infrastructure [2] : Local Database   : Not Replicated (Use
                              :         :                                :                  : 'dsreplication enable' to
                              :         :                                :                  : enable replication)

[1] You can update the integration using the 'manage-suffix update' sub-command.
[2] The suffix is required to integrate with EUS, Database Net Services or EBS.
[3] The suffix can integrate its contents
```

17.2.4 Modifying a Suffix Configuration

You can use the `manage-suffix update` command to modify an integrated suffix configuration. You can use either the interactive or non-interactive CLI.

This section contains the following topics:

- Modifying a Suffix Configuration Using the Non-Interactive CLI Mode
- Modifying a Suffix Configuration Using the Interactive CLI Mode

17.2.4.1 Modifying a Suffix Configuration Using the Non-Interactive CLI Mode

You can modify an integrated suffix configuration using the non-interactive CLI mode. The example described in this section use the `manage-suffix update` command to change the integration property from EUS to generic, which used for integrating either Oracle Database or E-Business Suite. The change is made for both `dc=suffix1` and `dc=suffix2`.

Run the `manage-suffix` update command as follows:

```
manage-suffix update \
          --baseDN dc=suffix1 \
          --baseDN dc=suffix2 \
          --integration generic \
          --hostname host1.local \
          --port 4444 \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile ****** \
          --trustAll \
          --no-prompt
```

17.2.4.2 Modifying a Suffix Configuration Using the Interactive CLI Mode

You can modify an integrated suffix configuration using the interactive CLI mode. The example described in this section use the `manage-suffix update` command to change the integration property from EUS to generic, which used for integrating either Oracle Database or E-Business Suite. The change is made for both `dc=suffix1` and `dc=suffix2`.

Run the manage-update command. For example:

```
$ manage-suffix update -h localhost -p 4444 -D "cn=directory manager" -j pwd-file

Reading Configuration ..... Done.

Choose the suffixes (separated by commas) to be updated.

    1)  cn=nointegrated
    2)  dc=suffix1
    3)  dc=suffix2
    4)  All

    c)  cancel

Enter one or more choices separated by commas: 2,3

Specify the Oracle components with which the suffixes can integrate.

    1)  Do not update the integration with Oracle components
    2)  No Integration
    3)  Generic: Database Net Services and EBS (E-Business Suite)
    4)  EUS (Enterprise User Security), Database Net Services and EBS
        (E-Business Suite)

    c)  cancel

Enter choice [1]: 3

Choose the network groups (separated by commas) that must expose the suffixes.

    1)  Do not update the network groups
    2)  network-group
    3)  network-group2
    4)  All
    5)  Create a new network group

    c)  cancel

Enter one or more choices separated by commas [1]:

Updating Oracle Integration ....... Done.
```

17.2.5 Deleting a Suffix Using `manage-suffix`

You can use the `manage-suffix delete` command to remove a suffix and all of its data. You can use the non-interactive CLI or the interactive CLI.

This section contains the following topics:

- Deleting a Suffix Using the Non-Interactive CLI Mode
- Deleting a Suffix Using the Interactive CLI Mode

17.2.5.1 Deleting a Suffix Using the Non-Interactive CLI Mode

You can delete a suffix using the non-interactive CLI mode.

Run `manage-suffix delete` with the `baseDN` argument. For example:

```
manage-suffix delete \
          --baseDN dc=nointegration \
```

```
--hostname host1.local \
--port 4444 \
--bindDN cn=Directory\ Manager \
--bindPasswordFile ****** \
--trustAll \
--no-prompt
```

17.2.5.2 Deleting a Suffix Using the Interactive CLI Mode

You can delete a suffix using the interactive CLI mode.

Run the `manage-suffix delete` command. For example:

```
$ manage-suffix delete -h localhost -p 4444 -D "cn=directory manager" -j pwd-file

Reading Configuration ..... Done.

Choose the suffixes (separated by commas) to be deleted.

    1)  cn=nointegrated
    2)  dc=suffix1
    3)  dc=suffix2
    4)  All

    c)  cancel

Enter one or more choices separated by commas: 1

You have chosen to delete the suffix 'cn=nointegrated'.
Once deleted, the data contained in the suffix will be permanently removed.

Do you want to continue? (yes / no) [no]: yes

Deleting suffix 'cn=nointegrated' ...... Done.
```

The non-integrated suffix `dc=nointegration` is deleted, and its local database workflow element `userRoot2` is also deleted. In these examples, local database workflow element `userRoot2` will also be deleted if `dc=nointegration` is the only base DN defined in it.

17.3 Managing the Server Configuration Using OUDSM

The Configuration tab of each server instance in OUDSM enables you to modify elements of the server configuration.

For additional information about managing the configuration that is specific to a proxy server instance, see Configuring Proxy, Distribution, and Virtualization Functionality

This section provides an overview of the tasks that can be performed on the Configuration tab in OUDSM, and covers the following topics:

- Understanding How to Select a Configuration View
- Using Shortcuts to Configure Objects Using OUDSM
- Configuring Suffixes Using OUDSM
- Configuring Workflow Elements Using OUDSM
- Configuring Workflows Using OUDSM
- Configuring Connection Handlers Using OUDSM

- Configuring Network Groups Using OUDSM
- Modifying the General Server Configuration Using OUDSM

17.3.1 Understanding How to Select a Configuration View

There are two separate views of the server configuration in the Configuration tab.

They are:

- **Naming Contexts.** This is the default view, and shows the server configuration in terms of the naming contexts or suffixes configured on that server instance.
- **Core Configuration.** This view displays the server configuration in terms of the workflows, workflow elements and server extensions configured on that server instance.

The configuration view that you select determines the items that are available under the **Create** menu.

17.3.2 Using Shortcuts to Configure Objects Using OUDSM

You can use shortcuts such as **Create Like** icon and to create new components with the same configuration and **Create** icon to create a similar type of component that is already selected.

When you create server components using OUDSM, you can duplicate an existing component using the **Create Like** icon. When you select a component on the configuration tab and click **Create Like**, a new component with the same configuration is created. You can then edit the properties of the new component to suit your requirements.

You can also use the **Create** icon to create the same type of component as the one you have selected. For example, if you select LDAP Connection Handler in the left hand menu, and click **Create**, a new, unconfigured LDAP connection handler is created.

Right-clicking on a component in the left hand menu provides a list of actions related to that component. For example, if you right-click **LDAP Connection Handler**, a drop-down menu is displayed, enabling you to create a new LDAP connection handler, duplicate that LDAP connection handler, or delete the connection handler.

17.3.3 Configuring Suffixes Using OUDSM

You can configure suffixes or naming contexts, using OUDSM.

For information about using `dsconfig` to configure suffixes, see Configuring Suffixes with `dsconfig`.

This section contains the following topics:

- Creating a Suffix
- Displaying and Editing Suffix Properties
- Deleting a Suffix

17.3.3.1 Creating a Suffix

You can create one or more suffixes using the OUDSM interface.

Perform the following steps to create a suffix:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Naming Contexts** view.
4. From the **Create** menu, select **Local Naming Context.**
5. In the Naming Context region, perform the following steps:
   a. In the **Base DN** field, type a name for the suffix that you want to create.
   b. From the Directory Data Options group, select one of the following options for populating the suffix with data:

      Only Create Base Entry creates the database along with the base entry of the suffix. Any additional entries must be added after suffix creation.

      Leave Database Empty creates an empty database. The base entries and any additional entries must be added after suffix creation.

      Import Generated Sample Data populates the suffix with sample entries.

      Specify the number of entries that should be generated in the **Number of User Entries** field. You can import a maximum of 30,000 sample entries through OUDSM. If you want to add more than 30,000 entries, you must use the `import-ldif` command.
6. In the Oracle Components Integration region, select one of the following option to enable the new suffix:
   - **No Specific Integration**: Select this option, if you do not want to integrate the naming context with Oracle components.
   - **Enable for Enterprise User Security (EUS)**:

     To enable a suffix for EUS, you must have at least one LDAP listener with SSL enabled, in addition to the administration listener. The suffix must contain at least one entry (in other words, you must *not* have selected "Leave Database Empty" in the previous step).

     When you select EUS, in addition to creating this suffix, two suffixes are created automatically: "`cn=oracleschemaversion`" and "`cn=oraclecontext.`" An EUS workflow element is also added in front of the Local Backend workflow element. Further, a DN renaming workflow element for "`cn=schema`" is added, so that it can be accessed using the "`cn=subschemasubentry`" DN.
   - **Enable for Oracle Database Net Services**: Select this option if you want the naming context to store the Database Connect Identifiers.
7. In the Network Group region, attach the suffix to at least one network group by performing the following steps:
   - To attach the suffix to an existing network group, select **Use Existing** and select the required network group from the list.
   - To attach the suffix to a new network group, select **Create New** and then in the **Name** field, type a name for the network group you want to create.

   You can attach several network groups to the same suffix.
8. In the Workflow Element region, attach the suffix to the workflow element by performing either of the following steps:
   - To attach the suffix to an existing workflow element, select **Use Existing** and then select the required workflow element from the list.

- To attach the suffix to a new workflow element, select **Create New** and then in the **Name** field, type a name for the workflow element you want to create. You can create a Local DB Workflow Element or a Local LDIF Workflow Element.

9. Click **Create.**

   The following confirmation message is displayed:

   ```
   Configuration created successfully.
   ```

   You can configure the tombstone entry purge interval and the tombstone entry lifetime after creating the suffix, in the Local Backend workflow element configuration.

17.3.3.2 Displaying and Editing Suffix Properties

In the Naming Contexts view, the Configuration tab displays all of the suffixes that have been configured on the server.

To display the properties of a configured suffix, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Naming Contexts** view.
4. Expand the **Naming Contexts** element.
5. Click the suffix whose properties you want to display.

   The suffix properties are displayed in the right hand pane.

6. Make any required changes to the suffix configuration.

   You can change the network group to which this suffix is attached, and enable the suffix for Enterprise User Security (EUS) or Enable for Oracle Database Net Services.

   > **Note:**
   >
   > If the Oracle Components Integration option was previously configured for the **Enable for Enterprise User Security (EUS)** or the **Enable for Oracle Database Net Services** options and if you have made changes in the Oracle Components Integration region, the **Configuration Required** dialog box appears. Depending on the option you choose, select one of the following:
   >
   > - **Keep Oracle Context**: Select this option, if you want to keep the naming context for EUS and Oracle Database Net Service.
   > - **Delete Oracle Context**: Select this option, if you want to delete the naming context for EUS and Oracle Database Net Service.

   Click **Apply** to save your changes.

17.3.3.3 Deleting a Suffix

In the Naming Contexts view, the Configuration tab displays all of the suffixes that have been configured on the server.

To delete a suffix, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Naming Contexts** view.
4. Expand the **Naming Contexts** element.
5. Select the suffix that you want to delete.
6. Click the **Delete configuration**  .

17.3.4 Configuring Workflow Elements Using OUDSM

A workflow element is the key building block of a workflow process. Workflow elements define how client requests that are sent to the server are treated.

In a deployment that includes a proxy server, workflow elements are configured for load balancing or distribution. In a deployment that does not include a proxy server, workflow elements are configured directly for each back end.

For information about using `dsconfig` to configure workflow elements, see Configuring Workflow Elements Using `dsconfig`.

The following sections describe how to configure workflow elements using OUDSM.

- Creating a Workflow Element
- Displaying and Editing Workflow Element Properties
- Deleting a Workflow Element

17.3.4.1 Creating a Workflow Element

You can create a workflow element using OUDSM.

Perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.
4. From the **Create** menu, select **Workflow Element** and select the type of workflow element that you want to create.

   For more information about the various workflow element types, see Understanding Workflow Elements.
5. When the Create page displays for the selected workflow element, configure the properties on that page.

> **Note:**
>
> The properties that you must configure will depend on the type of workflow element that you are creating.

All workflow elements require the following basic properties to be configured:

Name. Enter a name for the workflow element.

Enabled. When you create a workflow element, it is enabled by default. Clear this item to disable the workflow element.

In addition, you must configure relevant properties for each corresponding workflow element type depending on the workflow element that you are creating. For more information about the properties to configure for each of the following workflow element, see Configuring Properties of Workflow Elements.

To create the Join Workflow Element, use the Create Join Workflow Element wizard as follows:

a. Configure the following **General** properties and then click **Next**:

b. Configure the following **Primary Participant** properties and click **Next**.

c. The **Secondary Participant** page is displayed, and it contains a menu with options that enable you to **View**, **Create**, **Modify**, or **Remove** participants.

   To add one or more secondary participants, click **Create** and configure the properties on the Create Secondary Participant page. These properties are essentially the same as those you configured for the **Primary Participant**, except for the following:

   When you are finished adding participants, click **Next**.

d. Configure the following **Participant Relations Properties** to define Join Rule relations and to view or move Bind Participants. You can also view the join relations between participants in a tree structure. When you are finished, click **Next**.

e. Use the following **Network Group** properties to associate this workflow element with a network group. When you are finished, click **Next**.

f. When the Summary page is displayed, click back through all of the pages to review the property settings. If necessary, make any necessary changes.

g. If you are satisfied with the configuration, click **Create** to create the Join workflow element.

> **Note:**
>
> For more information about the each preceding Join workflow element properties, see Configuring Properties of Workflow Elements.

6. Click **Create.**

   The following confirmation message is displayed:

```
Workflow Element created successfully.
```

17.3.4.1.1 Configuring Properties of Workflow Elements

Each workflow element has properties associated with it that you must configure while creating that specific workflow element.

This section lists the properties for each workflow element type that are of relevance.

- **DN Renaming Workflow Element**

Property Name	Description
Client Base DN	Specify the base DN that is used by the client application.
Source Base DN	Specify the base DN that is stored in the LDAP server.
Next Workflow Element	Select the workflow element that should be next in the workflow.
Attribute White List	Click **Add** to select the list of attributes that contain DNs and must be transformed by the renaming operation.
Attribute Black List	Click **Add** to select the list of attributes that contain DNs but must *not* be transformed by the renaming operation.

- **EUS Workflow Element**

Property Name	Description
EUS Realm	Enter the part of the DIT to which the EUS workflow element applies.
Next Workflow Element	Select the workflow element that should be next in the workflow.
Server Type	Select the server containing the EUS user entries.
Password Attribute	Enter the attribute type that should be used to hold the EUS user passwords.

- **EUS Context Workflow Element**

Property Name	Description
EUS Context	Enter the DN that contains the Oracle Context. The Oracle Context is a top-level directory entry that contains the data used by any installed Oracle product that uses the directory.
EUS Administrator	Enter the DN of the administration user. This user will be the uniquemember of the groups created in Oracle Context.
Next Workflow Element	Select the workflow element that should be next in the workflow.

- **Kerberos Authentication Provider Workflow Element**

Property Name	Description
Realm	Specify the realm to be used for Kerberos authentication. If you do not specify a realm, then the server attempts to determine the realm from the underlying system configuration.
Principal Name Attribute	Click **Select** and specify the Principal Name Attribute.
KDC Address	Specify the Key Distribution Center (KDC) server address.

- **Local DB Workflow Element**

Property Name	Description
Writability Mode	Specify whether the back end associated with this workflow element should process write operations.
Base DN	Specify one or more base DNs for the data that is handled by the back end.
Database Properties	Specify any specific properties for the database. For a detailed list of these properties, and their values, see "DB Local Backend Workflow Element" in the *Configuration Reference for Oracle Unified Directory*.

Property Name	Description
Tombstone Configuration	Specify how tombstone entries should be handled for the database. For a detailed list of these properties, and their values, see "DB Local Backend Workflow Element" in the *Configuration Reference for Oracle Unified Directory*.
Index Properties	Specify the following parameters: – **Index Subtrees**: Enable or disable the check box to indicate whether the back end should index subtrees to maintain subtree specific data retaining information on direct and indirect children entries of each parent entry. – **Local DB Index**: Specify the local DB index configuration for the database. For a detailed list of these properties, and their values, see "DB Local Backend Workflow Element" in the *Configuration Reference for Oracle Unified Directory*. – **Local DB VLV Index**: Specify the local DB VLV index configuration for the database. For a detailed list of these properties, and their values, see "DB Local Backend Workflow Element" in the *Configuration Reference for Oracle Unified Directory*.

- **Local LDIF Workflow Element**

Property Name	Description
Writability Mode	Specify whether the back end associated with this workflow element should process write operations.
Base DN	Specify one or more base DNs for the data that is handled by the back end.
Private Backend	Specify whether the back end should be considered a private back end, which indicates that it is used for storing operational data rather than user-defined information.
LDIF File	Enter the path to the LDIF file containing the data for this back end.

- Pass Through Authentication Workflow Element

Basic Properties

Property Name	Description
User Provider Workflow Element	Select the workflow element providing the requested user entry.
Authentication Provider Workflow Element	Select the workflow element providing the authentication service for the user entry. For example, you can use Kerberos Authentication Provider workflow element or Local DB workflow element as the authentication provider.

Advanced Properties

Property Name	Description
Password Attribute	Click **Select** and specify the password attribute.
Save Password on successful bind	Enable the check box, if you want the Authentication Provider workflow element to trigger a copy of the password to the User Provider workflow element.
Pass Through Authentication Suffix	Specify the virtual suffix that is exposed through the PTA workflow element.

Property Name	Description
User Suffix	Specify the suffix that contains the user entries on the User Provider workflow element.
Authentication Suffix	Specify the suffix that contains the authentication entries on the Authentication Provider workflow element.

Pass Through Authentication Join Rule Properties

Property Name	Description
Auth Entry Property	Specify the authentication entry property associated with the user entry.
User Entry Property	Specify the user entry property associated with the authentication entry.

For more information, see Understanding Pass-Through Authentication.

- **Local Memory Workflow Element**

Property Name	Description
Base DN	Specify one or more base DNs for the data that is handled by the back end.

- **RDN Changing Workflow Element**

Property Name	Description
Next Workflow Element	Select the workflow element that should be next in the workflow.
Object Class	Select the object class type for RDN changing operation.
Source RDN Attribute	Select the original RDN attribute name from the source directory to be replaced or renamed in Oracle Unified Directory.
Client RDN Attribute	Select the new RDN attribute name to be used in Oracle Unified Directory.
Replace RDN Value	Specify whether the original RDN value should be replaced by the new RDN value. It is enabled by default.
DN Attributes	Click **Add** to select the list of attributes with DNs on which to perform RDN renaming.

- **Transformations Workflow Element**

Property Name	Description
Next Workflow Element	Select the workflow element that should be next in the workflow.
Entry Matching Filter	This is an LDAP filter. If you select this option, then entries will be transformed only if they match this LDAP Filter.
Entry Parent Suffixes	*Optional* You can specify a list of suffixes to restrict applying transformation to entries under specific subtrees. If you specify this option, then the entries will be specified only if they are in subtrees rooted at any of these suffixes.
Excluded Operations	If you specify this option, then the entries will not be transformed during any of the specified operations.
Transformations	The list of transformations that the Transformations Workflow Element will process. The order in which transformations are listed here does not guarantee the order in which these transformations will be applied when processing a request at runtime.).

- **Join Workflow Element**

General Properties

Property Name	Description
Name	Enter a name for the workflow element. For example, `we-join`
Enabled	Option is enabled by default, indicating the workflow element is enabled. If necessary, you can disable this element later by returning to this page and clearing the box.
Join Suffix	Enter the virtual DN to be exposed by the Join workflow element. For example, `dc=join`
DN Attributes	*Optional*. Click **Add** to create a list of attributes (such as `manager`, `memberof`, or `uniquemember`) with DNs on which to perform the join.
Populate the virtual attribute '`joinedentrydn`' in retrieved entries	*Optional*. Enable this box to populate the virtual attribute with the entries from secondary participants that were used to form the consolidated entry **Note**: This information is useful when troubleshooting Join issues.

Primary Participant Properties

Property Name	Description
Participant Name	Enter the name of the participant that will contribute information to form the combined joined entry. For example, `jp-p1`
Participant Workflow Element	Enter the name of the workflow element that the primary participant will use to attach itself. For example, `we-proxy1`
Participant DN	Enter the suffix DN of the participating workflow element or a subtree of that element. For example, `dc=com1`
Enabled Operations	*Optional*. Click the menu button to view a list of operations, which include: Add, Bind, Compare, Delete, Modify, Modify DN, and Search. – Select one or more boxes to enable operations. – Clear the boxes to disable operations.
Criticality	Specify one of the following criticality flags for the join workflow element: – **true** (default): Indicates the participant is critical. If the participant fails to return a result due to an operation error, then the overall operation fails and an error message results. – **false**: Indicates that a failure to perform an operation in the participant is not critical to the overall result. – **partial**: Indicates the participant is partially critical. If the participant fails to return a result due to an operation error, then the application can notify its own users that partial results were obtained, the Join workflow element returns partial results, but also returns an error message.

Property Name	Description
Join Condition	This field is blank by default, indicating that no join condition is defined. All entries that satisfy the original search filter will be considered for a join. To restrict the entries to be joined, click **Define** to access the Filter Builder dialog where you configure a filter: 1. Select an attribute name from the left menu. 2. Select a matching rule from the middle menu. 3. Enter a value to match. 4. Click **Add** to create another filter. 5. When you are finished creating filters, click **OK**. Entries that do not satisfy the specified conditions are returned as is, with no join done on them. **Note**: For information, see Understanding the Join Condition.
Attribute Storage	Enable one of the following to specify which attributes the Join participant can store on the target directory: – **All attributes are storable** (default): All attributes can be stored. – **Only the selected attributes are storable**: Click **Add** and then click the search icon  to select one or more attributes from the Attribute Picker dialog. Only the selected attributes can be stored. – **All except the following attributes are storable**: Click **Add** and then click the search icon  to select one or more attributes from the Attribute Picker dialog. All attributes can be stored except for the selected attributes.
Attribute Retrieval	Enable one of the following to specify which attributes the Join participant can retrieved from the target directory: – **All attributes are retrievable** (default): All attributes can be retrieved. – **Only the selected attributes are retrievable**: Click **Add** and then click the search icon  to select one or more attributes from the Attribute Picker dialog. Only the selected attributes can be retrieved. – **All except the following attributes are retrievable**: Click **Add** and then click the search icon  to select one or more attributes from the Attribute Picker dialog. All attributes can be retrieved except for the selected attributes.

Secondary Participant Properties

Property Name	Description
Joiner Type	Click the button and select one of the following joiner types from the menu: – **Many to one**: – **One to many** – **One to one** (default) – **Shadow** For a description of these join relationships, see Understanding Supported Joiner Types.

Participant Relations Properties

Property Name	Description
Define Join Rule Relations	Click **Define** to open the Filter Builder dialog, where you can specify filter criteria for a join rule. **Note**: You might have to enlarge the browser window to access this button. To specify the filter criteria: 1. Select the leftmost menu button to choose an attribute name for the first participant. 2. Click the next button to choose a matching rule. 3. In the next field, enter or choose the other participant name. 4. Select the rightmost button to choose an attribute name for the second participant. Click the plus sign icon to add additional filter rules. When you are finished, click **OK**.
Participant Relations	Use this area to view the join relations between participants.
Bind Participants	View or move the participants up or down in the table.

Network Group Properties

Property Name	Description
Associate this workflow element with a Network Group	Enable this box to associate this workflow element with a network group and workflow. When you enable the box, the other features on this page become active.
Select the network group to which the workflow element must be attached	Enable the **network-group** box to attach the workflow element to that group. Click **Create Network Group** to open the Create Network Group dialog. Enter a new name and click **Create**.
Create New Workflow	Enter a name in the field to create a new workflow.

17.3.4.2 Displaying and Editing Workflow Element Properties

After you have created a workflow element, you can view or modify the properties of an existing workflow element.

Perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.
4. Expand the **Core Configuration** element.
5. Expand the **Workflow Elements** element.
6. Click the workflow element that you want to view or modify.

   The properties of the workflow element are displayed in the right hand pane
7. The properties that you can edit depend on the type of workflow element that is configured.
8. Click **Apply** to save your changes.

17.3.4.3 Deleting a Workflow Element

To delete an existing workflow element using OUDSM, perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.
4. Expand the **Core Configuration** element.
5. Expand the **Workflow Elements** element.
6. Click the workflow element that you want to delete and click the **Delete configuration**  .
7. Click **OK** to confirm the deletion.

17.3.5 Configuring Workflows Using OUDSM

A workflow is defined by a naming context, or suffix, and a workflow element that define how Oracle Unified Directory should handle an incoming request. A workflow must be registered with at least one network group, but can be attached to several network groups.

The following sections describe how to configure workflows using OUDSM:

- Creating a Workflow
- Displaying and Editing Workflow Properties
- Deleting a Workflow

For more information about workflows, workflow elements and the other components of Oracle Unified Directory, see Understanding Oracle Unified Directory Components.

For information about configuring workflows using `dsconfig`, see Configuring Workflows Using `dsconfig`.

17.3.5.1 Creating a Workflow

To create a workflow using OUDSM, perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.
4. From the **Create** menu, select **Workflow.**
5. In the Workflow Properties region, enter the following information:
   a. In the **Name** field, type a name for the workflow that you want to create.
   b. Select the **Enabled** check box if you want this workflow to be enabled.

      Deselect this check box if you do not want to enable the workflow at this stage.

6. In the **Base DN** field, enter the naming context that will be accessible through this workflow.
7. Select the **Workflow Element** with which this workflow should be associated.

   The workflow element must already exist before you create the workflow.
8. Select **True**, **False**, or **Partial** depending on whether the workflow is critical enough to fail a search operation involving multiple workflows and if the operation fails on this workflow.
9. Select the **Use Virtual ACIs** check box if you want to define a different storage repository for the ACI data associated to all entries managed by the workflow.
10. If the **Use Virtual ACIs** check box is selected then specify the name of the stripe to be used in the Virtual ACI storage to maintain ACI data for this workflow.
11. Click **Create.**

    The following confirmation message is displayed:

    ```
    Workflow created successfully.
    ```

17.3.5.2 Displaying and Editing Workflow Properties

In the Core Configuration view, the Configuration tab displays all of the workflows and workflow elements that have been configured on the server.

To display the properties of a configured workflow, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.
4. Expand the **Workflows** element.
5. Click the workflow whose properties you want to display.

   The workflow properties are displayed in the right hand pane.
6. Make any required changes to the suffix configuration.

   You can disable the workflow, or change the workflow element with which this workflow is associated.

   Click **Apply** to save your changes.

17.3.5.3 Deleting a Workflow

You use OUDSM to delete a workflow only if that workflow is not referenced by any network group.

To delete a workflow:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. If the workflow is referenced by a network group, modify the properties of the network group to remove that workflow.

   For more information, see Modifying a Network Group.
3. Select the **Configuration** tab.

4. Select the **Core Configuration** view.

   For more information, see Understanding How to Select a Configuration View.

5. Expand the **Workflows** element.

6. Select the workflow that you want to delete and click the **Delete configuration**  .

7. Click **OK** to confirm the deletion.

17.3.6 Configuring Connection Handlers Using OUDSM

Connection handlers are responsible for accepting connections from clients, reading and parsing requests submitted by the clients, ensuring that they are processed by the server, and sending the corresponding responses back to the client. A connection handler manages all communication with the client and therefore needs to implement support for the associated protocol.

The following sections describe how to configure connection handlers using OUDSM:

- Creating a Connection Handler
- Modifying a Connection Handler
- Deleting a Connection Handler

For information about configuring connection handlers using `dsconfig`, see Configuring Connection Handlers Using `dsconfig`.

17.3.6.1 Creating a Connection Handler

To create a connection handler using OUDSM, perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. From the **Create** menu, select **Connection Handler**.
4. Select the type of connection handler that you want to create:
   - **LDAP.** This connection handler is used to interact with clients using LDAP. It provides full support for LDAPv3 and limited support for LDAPv2.
   - **LDAPS.** This connection handler is used to interact with clients using LDAP over SSL.
   - **LDIF.** This connection handler is used to process changes in the server using internal operations.
   - **JMX.** This connection handler allows interactions with clients using the Java Management Extensions (JMX) framework and the Remote Method Invocation (RMI) protocol.
   - **SNMP.** This connection handler is used to process SNMP requests to retrieve monitoring information described by MIB 2605. The supported SNMP protocol are SNMP V1, V2c, and V3.
5. Enter the properties to configure the connection handler in the right hand pane.

   The configurable properties will depend on the type of connection handler you have selected. For a comprehensive list of all configurable properties, and their allowed values, see "The Connection Handler Configuration" in the *Configuration Reference for Oracle Unified Directory*.

6. When you have configured the required properties for your specific connection handler type, click **Create**.

   The following confirmation message is displayed:

   ```
   Connection Handler created successfully.
   ```

17.3.6.2 Modifying a Connection Handler

To view or modify connection handler properties using OUDSM, perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Connection Handlers** element.
5. Click the connection handler whose properties you want to modify.

   The properties are displayed in the right hand pane.

   For a comprehensive list of all configurable properties, and their allowed values, see "The Connection Handler Configuration" in the *Configuration Reference for Oracle Unified Directory*.
6. Change the required properties and click **Apply**.

17.3.6.3 Deleting a Connection Handler

To delete an existing connection handler using OUDSM, perform the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Connection Handlers** element.
5. Click the connection handler that you want to delete and click the **Delete configuration**  .
6. You are prompted to confirm the deletion. Click **OK**.

17.3.7 Configuring Network Groups Using OUDSM

Network groups are the entry point of all client requests that are handled by an Oracle Unified Directory server. The properties of a network group indicate how client requests are directed.

The following sections describe how to configure network groups using OUDSM:

- Creating a Network Group
- Modifying a Network Group
- Deleting a Network Group

For information about configuring network groups using `dsconfig`, see Configuring Network Groups Using `dsconfig`.

17.3.7.1 Creating a Network Group

To create a network group using OUDSM, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. From the **Create** menu, select **Network Group**.
4. Configure the network group by using the properties available in the right-hand pane. For more information about these properties, see Configuring Properties of a Network Group.
5. When you have configured the required properties for the network group, click **Create**.

   The following confirmation message is displayed:

   ```
   Network Group created successfully.
   ```

17.3.7.1.1 Configuring Properties of a Network Group

There are several properties associated with a network group that you must configure while creating a network group using OUDSM.

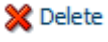
This section lists that these properties in details.

- **Name.** Enter a name for the network group.
- **Enabled.** Select or deselect this check box to enable or disable the network group. If you disable a network group, then no client requests can be handled by that network group. If you disable the only configured network group, then you effectively stop client applications from accessing the back end.
- **Priority.** If you have multiple network groups, specify a priority for this network group. Client requests are handled by the network group with the highest priority, for which the criteria are met. The highest priority a network group can have is 0.
- **Workflow.** Click the **Add** (

) to add one or more workflows that can be accessed through this network group.
- **Root DSE to Expose.** Select the Root DSE that you want this network group to expose. You can expose the Root DSE of the local server, the Root DSE stored in a remote server, or the Root DSE defined in a local file.

  Click **Other** and select one of the following options:

Option	Description
`Root DSE Defined in LDIF File`	Enter the path of the LDIF file containing the Root DSE. The server must have access to this file.
`Root DSE of a Remote Server`	Enter the following parameters: **Host Name**: Enter the host name of the remote server. **Ports Available**: Enter the LDAP port, LDAPS port, or LDAP and LDAPS ports of the remote server. **Trust All**: Select this check box to trust all the certificates presented by the remote server. **Trust Manager**: Select the trust manager that the server will use when connecting to the LDAPS ports of the remote server to forward requests.

- **Security Mandatory.** Select this option if you require clients to use a secure connection to access this network group. By default, a secure connection is not required.
- **Allowed auth method.** Specify the authentication method/s that are allowed between the client and the network group.
- **Allowed protocol.** Specify the protocol/s that are allowed for client connections. If you do not specify a protocol, all protocols are allowed.
- **Allowed BindDN.** Click the Add to add one or more bind DNs that are allowed to connect to this network group. Click the **Delete** () to remove the bind DNs that should not be accepted by the network group.
- **Allowed Client.** Click the Add to add one or more clients that are authorized to access this network group. Clients can be expressed by their IP addresses or names, or by a subnet mask. If no allowed client list is provided, all clients are allowed, unless they are specifically listed on the denied client list.
- **Denied Client.** Click the Add to add one or more clients that are prohibited from accessing this network group. Clients can be expressed by their IP addresses or names, or by a subnet mask. If no denied client list is provided, all clients are allowed, unless a limitation is set using the allowed client list.
- **QoS Policy.** Select a quality of service policy for this network group. For more information, see Creating a Network Group Quality of Service Policy.

17.3.7.2 Modifying a Network Group

To view or modify the properties of a network group using OUDSM, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Network Groups** element.
5. Select the network group whose properties you want to modify.

   The properties of the network group are displayed in the right hand pane
6. Change the required properties and click **Apply**.

   For an explanation of each of the configured properties, see Configuring Properties of a Network Group.

17.3.7.3 Deleting a Network Group

To delete an existing network group using OUDSM, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Network Groups** element.
5. Click the network group that you want to delete and click the **Delete configuration** .
6. You are prompted to confirm the deletion. Click **OK**.

17.3.8 Modifying the General Server Configuration Using OUDSM

You can modify the general server configuration using OUDSM by accessing the directory server from OUDSM and then by modifying the General Server properties.

To modify elements of the general server configuration using OUDSM, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **General Configuration** element.

   The properties are displayed in the right hand pane.
4. You can modify the following properties:
   - **Root DSE Properties**
   - **Work Queue Properties**
   - **Access Control Groups**
   - **Data Encryption Properties**
   - **Number of Worker Threads**
   - General Server Properties

   For more information about the preceding properties, see Managing the General Server Configuration Properties.
5. Click **Apply** to save your changes.

17.3.8.1 Managing the General Server Configuration Properties

There is a comprehensive list of configurable properties. This section describes the general server configuration properties that you can modify using OUDSM.

General Server Properties

- **Default Password Policy**: Specify the name of the password policy, if the entries do not have an alternate password policy,
- **Etime Resolution**: Select a resolution for operation elapsed processing time measurements. The default value is Milliseconds.
- **Idle Time Limit**: Specify the maximum duration a client connection may remain established since its last completed operation. If you specify `0 seconds` as the value, then no idle time limit is enforced.
- **Max Allowed Client Connections**: Specify the maximum number of client connections you want to establish at any given time. A value of 0 indicates that unlimited client connection is allowed.
- **Maintain Authenticated Users**: Select the check box, if you want the server to maintain authenticated users.
- **Reject Unauthenticated Requests**: Select the check box, if you want the directory server to reject any request (other than bind or StartTLS requests) received from a client that has not yet been authenticated, whose last authentication attempt was unsuccessful, or whose last authentication attempt used anonymous authentication.

- **Size Limit**: Enter a value to specify the maximum number of entries that can be returned to the client during a single search operation. A value of 0 indicates that no size limit is enforced. This is the default server wide limit, but it may be overridden on a per user basis using the `ds-rlim-size-limit` operational attribute.
- **Writability Mode**: Specify the type of write operations the Directory Server can process.

Root DSE Properties

- **Show Operational Attributes**: Select this check box, if you want all attributes in the root DSE to be treated like user (non operational) attributes (and therefore returned to clients by default) regardless of the Directory Server schema configuration.
- **Subordinate Base DNs**: Specify the set of base DNs used for singleLevel, wholeSubtree, and subordinateSubtree searches based at the root DSE.

Work Queue Properties

- **Number of Worker Threads**: Specify the number of worker threads to be used for processing operations placed in the queue. If the value is increased, the additional worker threads are created immediately. If the value is reduced, the appropriate number of threads are destroyed as the operations complete processing.
- Click the **Dynamically Handled by Server** check box, if you want the server to determine the number of worker threads at run time.
- **Maximum Work Queue Capacity**: Specify the maximum number of queued operations that can be in the work queue at any given time. If the work queue is already full and additional requests are received by the server then the server front end and possibly the client will be blocked until the work queue has available capacity.

Data Encryption Properties

- Check **Enabled** check box, to enable encryption.
- **Encryption Algorithm**: Select the algorithm value for encryption. The default value is AES_128.
- **Encrypted Attributes**: Define the attributes for encryption.

  If you enable Data Encryption, then you must add the attribute names.
- **Suffixes to Apply Encryption**: Define the suffix for encryption.
  - If you do not define a suffix, then encryption is applied to all available suffixes.
  - If you define a suffix, then encryption is only applied to the defined suffix.

> **Note:**
>
> Modifying data encryption configuration may cause the indexes to be invalid. If any of the selected attributes are indexed then you must rebuild the indexes for these attributes as described in rebuild-index.

For a comprehensive list of the configurable properties, and their allowed values, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

17.4 Managing Administration Traffic to the Server

Connection handlers are responsible for handling all interaction with client applications, including accepting connections, reading requests, and sending responses.Oracle Unified Directory includes a special connection handler, the administration connector, to manage administration traffic to the server.

The administration connector enables the separation of user traffic and administration traffic to simplify monitoring, and to ensure that administrative commands take precedence over commands that manipulate user data.

This section describes how administration traffic is handled, and covers the following topics:

- Understanding the Administration Connector
- About Administrative Suffixes Access
- Configuring the Administration Connector
- Modifying Key Manager and Trust Manager Properties for the Administration Connector

17.4.1 Understanding the Administration Connector

The administration connector is based on the LDAP protocol and uses LDAP over SSL by default. All command-line utilities that access the administrative suffixes use the administration connector.

This includes the following commands:

- `backup`
- `dsconfig`
- `dsreplication`
- `export-ldif`
- `import-ldif`
- `manage-account`
- `manage-tasks`
- `restore`
- `status`
- `stop-ds`
- `uninstall`

The administration connector is always present and enabled. You cannot disable or delete the connector but you can use `dsconfig` to manipulate the following properties of the connector:

- `listen-address`. The address on which the server listens for administration traffic.
- `listen-port`. The default port of the administration connector is 4444. You can change this port during setup if required. If you use the default port, you do not need to specify a port when running the administration commands (the default port is assumed). If you change the port, you must specify the new port when running the administration commands.

  If you have multiple directory server instances running on the same host, you will have specified multiple separate administration listen ports during setup. In this case, for the

server instances whose administration connectors do not use the default listen port (4444), you will need to specify the port when running the administration commands.

- **Security-related properties.** Traffic using the administration connector is always secured. As with the LDAPS connection handler, the administration connector is configured with a self-signed certificate (`admin-cert`) during server setup. This self-signed certificate is generated the first time the server is started. You can manage the administration connector certificate using external tools, such as `keytool`.

  The security-related properties of the administration include the following:

  - `ssl-cert-nickname`
  - `ssl-cipher-suite`
  - `key-manager-provider`
  - `trust-manager-provider`

  When you run the administration commands, you are prompted about how you want to trust the certificate. If you run the administration commands in non-interactive mode, you must specify the `-X` or `--trustAll` option to trust the certificate, otherwise the command will fail.

17.4.2 About Administrative Suffixes Access

In general, direct LDAP access to the administrative suffixes (using the `ldap*` utilities) is discouraged, with the exception of the `cn=monitor` suffix. In most cases, it is preferable to use the dedicated administrative command-line utilities to access these suffixes.

If you must use the `ldap*` commands to access the administrative suffixes, you must use the administration connector port (with the `--useSSL` or `-Z` option). Using the administration connector ensures that monitoring data is not polluted and that server administration takes precedence over user traffic. The same restriction applies if you are accessing the administrative suffixes using an LDAP browser.

The *administrative suffixes* include the following:

- `cn=config`
- `cn=monitor`
- `cn=tasks`
- `cn=backups`
- `cn=ads-truststore`
- `cn=schema`
- `cn=admin data`

17.4.3 Configuring the Administration Connector

The example in this section displays the default properties of the administration connector, and changes the listen port of the connector to `5555`.

To change the default properties of the administration connector, follow these steps:

1. View the default properties of the administration connector, using the `dsconfig` command.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-administration-connector-prop
```

The output is similar to the following.

```
Property               : Value(s)
-----------------------:---------------
key-manager-provider   : Administration
listen-address         : 0.0.0.0
listen-port            : 4444
ssl-cert-nickname      : admin-cert
ssl-cipher-suite       : -
trust-manager-provider : Administration
```

2. Change the listen port, using the `dsconfig` command.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
 set-administration-connector-prop --set listen-port:5555
```

> **Note:**
>
> You must restart the server for changes to this property to take effect.

17.4.4 Modifying Key Manager and Trust Manager Properties for the Administration Connector

The administration connector is an LDAPS connector. As with all SSL-based connectors, the administration connector requires a key manager and trust manager.

Oracle Unified Directory provides a dedicated key manager and trust manager for the administration connector, which are enabled by default. You can change the properties of the default administration key manager and trust manager. For more information, see Configuring Key Manager Providers and Configuring Trust Manager Providers.

17.5 Configuring Commands As Tasks

You can use command-line utilities to schedule tasks that run within the directory server and perform their functions locally. These scheduled tasks support options used to connect to the directory server to interact with the task back end.

This section includes the following topics:

- About Commands That Can Schedule Tasks
- Controlling Which Tasks Can Run
- Scheduling and Configuring Tasks
- Managing and Monitoring Scheduled Tasks

17.5.1 About Commands That Can Schedule Tasks

Learn about commands that can schedule tasks.

The following utilities can schedule tasks:

- `import-ldif`
- `export-ldif`
- `backup`

- `restore`
- `stop-ds`
- `stop-ds --restart`
- `rebuild-index`
- `dsreplication purge-historical`

For a proxy server, only the `stop-ds` command can be scheduled to run as a task.

17.5.2 Controlling Which Tasks Can Run

You can control the tasks that can be run by setting the `allowed-tasks` advanced global configuration property. By default, all tasks supported by the tasks back end are allowed. To prevent a task from being run, remove its value from the `allowed-tasks` property.

For example, to prevent the server from being stopped using a task, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-global-configuration-prop --remove \
  allowed-task:org.opends.server.tasks.ShutdownTask
```

17.5.3 Scheduling and Configuring Tasks

The procedures in this section indicate how to schedule a task, how to configure task notification, and how to configure task dependencies. All of the examples in this section assume that the commands are being run on the local host, using the default administration port (4444), and the local certificate configuration.

If you run the commands remotely, you might need to specify the certificate parameters. For more information, see Managing Administration Traffic to the Server.

The following topics describe procedures to schedule and configure tasks:

- Scheduling a Task
- Scheduling a Recurring Task
- Configuring Task Notification
- Configuring Task Dependencies

17.5.3.1 Scheduling a Task

To schedule a task, invoke the required utility with the options used to connect to the directory server, an optional start time, and any options that will be used as arguments for the task execution.

If the `-t` or `--start` option is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time.

If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Schedule the `export-ldif` task to start at 12:15 on September 24th, 2009 as follows:

```
$ export-ldif -D "cn=directory manager" -j pwd-file \
  -l /ldif-files/example.ldif --start 20090924121500 -n userRoot
```

17.5.3.2 Scheduling a Recurring Task

You can schedule a recurring task using the recurring task option of the required utility.

This section contains the following topics:

- About Recurring Tasks Scheduling
- Configuring Recurring Tasks

17.5.3.2.1 About Recurring Tasks Scheduling

To schedule a recurring task, invoke the required utility with the options used to connect to the directory server, specifying the recurring task schedule, and any options that will be used as arguments for the task execution. The following commands can be scheduled as recurring tasks:

- `import-ldif`
- `export-ldif`
- `backup`
- `restore`
- `rebuild-index`
- `dsreplication purge-historical`

The `--recurringTask` option specifies a recurring task schedule that is used by the task scheduler to determine when and how often a recurring task should run. The pattern used to specify the schedule is based on UNIX `crontab`(5) scheduling patterns and rules and includes the following five integer pattern fields, separated by blank spaces:

- Minute [0,59]
- Hour [0,23]
- Day of the month [1,31]
- Month of the year [1,12]
- Day of the week [0,6] (with 0=Sunday)

Each of these patterns can be either an asterisk (meaning all valid values), an element, or a list of elements separated by commas. An element is either a number or two numbers separated by a dash (meaning an inclusive range).

The task scheduler spawns regular task iterations according to the specified schedule.

Schedule the task using the `--recurringTask` option.

17.5.3.2.2 Configuring Recurring Tasks

You can schedule recurring tasks that run automatically and repeatedly at specified time. This section describes example scenarios to configure recurring tasks.

The following command schedules a backup task to execute at the beginning of every hour.

```
$ backup -D "cn=directory manager" -j pwd-file --recurringTask \
  "00 * * * *" --backupDirectory /example/backup --backUpAll --backupID "Hourly Backup"
```

The following example shows an export task that is scheduled to run every 15 minutes, every Sunday.

```
$ export-ldif -D "cn=directory manager" -j pwd-file --recurringTask \
  "0,15,30,45 * * * 0" -l PATH/export-recurring.ldif -n userRoot
Recurring Export task  ExportTask-a614e45d-6ba5-4c29-a8e1-d518c20e46ab scheduled
successfully
```

17.5.3.3 Configuring Task Notification

The task scheduling options of a utility enable you to notify an administrator when a task completes or if an error occurs during the task's execution. To use the notification facility, an SMTP server must be configured for the directory server.

To configure the task notification, follow these steps:

1. Specify an SMTP server by setting the `smtp-server` global configuration property.

   The following command configures the SMTP server named `mailserver.example.com`:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     set-global-configuration-prop --set smtp-server:mailserver.example.com
   ```

2. Use the `completionNotify` and `errorNotify` options to specify the email address to which the task notification should be sent.

   The following command schedules a `backup` task and specifies that `admin@example.com` should be notified when the task completes, or when an error occurs:

   ```
   $ backup -D "cn=directory manager" -j pwd-file -a -d /tmp/backups \
     --start 20080924121500 --completionNotify admin@example.com \
     --errorNotify admin@example.com
   Backup task 20080924121500 scheduled to start Sep 24, 2008 12:15:00 PM SAST
   ```

17.5.3.4 Configuring Task Dependencies

Certain tasks might require that another task be completed before the task begins. The task dependency options of a utility enable you to specify that the task depends on another task, and what the task should do should the other task fail.

Schedule the task and specify the `dependency` and `failedDependencyAction`.

The following example schedules a backup task that depends on another task, and specifies that the backup should be canceled should the other task fail:

```
$ backup -D "cn=directory manager" -j pwd-file -a -d /tmp/backups \
  --start 2008102914530410 --dependency 20080924121500 \
  --failedDependencyAction cancel
Backup task 2008102914530410 scheduled to start Oct 29, 2008 14:53:04 PM SAST
```

17.5.4 Managing and Monitoring Scheduled Tasks

The `manage-tasks` utility can be used to obtain a list of scheduled tasks, to display task status, and to cancel scheduled tasks.

The following procedures provide examples of managing scheduled tasks:

- Viewing Information About Scheduled Tasks

- Canceling a Scheduled Task
- Canceling a Recurring Task

17.5.4.1 Viewing Information About Scheduled Tasks

To view information about scheduled tasks, follow these steps:

1. Display a summary of all scheduled tasks.

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n -s
ID                Type    Status
-------------------------------------------------
2008100912550010  Backup  Completed successfully
2008100912554710  Backup  Completed successfully
2008100912560510  Backup  Waiting on start time
2008100912561410  Backup  Waiting on start time
```

2. Display additional information on a particular task, specified by its task ID.

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n -i 2008100912550010

Task Details
------------------------------------------------------
ID                       2008100912550010
Type                     Backup
Status                   Completed successfully
Scheduled Start Time     Immediate execution
Actual Start Time        Oct 9, 2008 12:55:00 PM SAST
Completion Time          Oct 9, 2008 12:55:01 PM SAST
Dependencies             None
Failed Dependency Action  None
Email Upon Completion    None Specified
Email Upon Error         None Specified

Backup Options
----------------------------
Backup All        true
Backup Directory  ../backups

Last Log Message
------------------------------------------------------------------------------
[09/Oct/2008:12:55:01 +0200] severity="NOTICE" msgCount=4 msgID=10944795
message="The backup process completed successfully"
```

17.5.4.2 Canceling a Scheduled Task

You can cancel an existing scheduled task.

Run the `manage-tasks` utility with the `-c` or `--cancel` option.

The following command cancels a particular task, specified by its task ID:

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n -c 2008100912561410
```

17.5.4.3 Canceling a Recurring Task

You can cancel an entire recurring task, in which case both the recurring task and its next scheduled iteration are canceled. Alternatively, you can cancel only the next scheduled task iteration, in which case future recurring task iterations will be spawned by the task scheduler.

To cancel a recurring task, follow these steps:

1. Use the manage-tasks command to display the summary of scheduled tasks.

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n -s

 ID                                                  Type    Status
------------------------------------------------------------------------------
 Hourly Backup                                       Backup  Recurring
 Hourly Backup - Wed Jan 14 13:00:00 SAST 2009       Backup  Waiting on start time
```

2. Run the `manage-tasks` utility with the `-c` or `--cancel` option.

   a. Cancel the entire recurring task by specifying its task ID.

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n -c "Hourly Backup"
Task Hourly Backup canceled
```

   b. Cancel the next scheduled task by specifying its task ID.

```
$ manage-tasks -D "cn=directory manager" -j pwd-file -n \
  -c "Hourly Backup - Wed Jan 14 13:00:00 SAST 2009 "
Task Hourly Backup - Wed Jan 14 13:00:00 SAST 2009  canceled
```

17.6 Deploying and Configuring the DSML Gateway

The Directory Services Markup Language (DSML) is a SOAP-based mechanism that can communicate with directory servers using an XML-based representation instead of the LDAP protocol.

Oracle Unified Directory supports the use of DSML through a web application that acts as a DSML-to-LDAP gateway, in which clients communicate with the gateway using DSML, but the gateway communicates with the directory server through LDAP.

The following topics describe how to configure and deploy the DSML gateway:

- Deploying the DSML Gateway
- Confirming the DSML Gateway Deployment

17.6.1 Deploying the DSML Gateway

The DSML gateway can be deployed like any other web application, in most common application containers.

To deploy the DSML Gateway in Oracle WebLogic Server, you must perform the following steps:

1. Ensure that you have installed Oracle WebLogic Server, as described in Installing Oracle WebLogic Server.
2. Configure the WebLogic Server for the DSML Gateway as described in Configuring WebLogic Server for the DSML Gateway.
3. Deploy the DSML Gateway WAR file, as described in Deploying the DSML Gateway WAR File.

17.6.1.1 Configuring WebLogic Server for the DSML Gateway

To configure a WebLogic Server for the DSML Gateway, follow these steps:

1. Run the configuration wizard from the following location:

   OUD_BASE_LOCATION_HOME`/wlserver_10.3/common/bin/config.sh`

2. On the Welcome screen, select **Create a new WebLogic domain** and click **Next**.
3. On the Select Domain Source screen, accept the default selection (**Basic WebLogic Server Domain**) and click **Next**.
4. On the Specify Domain Name and Location screen, type a domain name and specify its location.

   A new WebLogic domain is created in this location. The DSML gateway will be deployed into this domain.
5. On the Configure Administrator User Name and Password screen, type a name and password for the user who will administer this domain.

   The password must be at least eight characters and must contain at least one number or special character. Confirm the password and click **Next.**

   Make a note of these details as you will need them to start or restart the WebLogic domain.
6. On the Configure Server Start Mode screen, select **Production Mode.**

   Select a valid JDK and click **Next.**
7. On the Optional Configuration screen, click **Next.**
8. On the Configuration Summary screen, verify the domain details and click **Create.**
9. On the Creating Domain Screen, click **Done.**
10. Set the Java options for the WebLogic Server.

    ```
    $ export JAVA_OPTIONS=-
    Djavax.xml.soap.MessageFactory=weblogic.xml.saaj.MessageFactoryImpl
    ```

    If you do not set the Java options, an error will be returned.
11. Set the `enforce-valid-basic-auth-credentials` flag in the configuration file of the WebLogic domain (*DOMAIN_HOME*`/config/config.xml`, where `DOMAIN_HOME` is the domain that you created in Step 4.

    For example, edit the file *OUD_BASE_LOCATION_HOME*`/user_projects/domains/base_domain/config/config.xml` by adding the following line to the `security-configuration` element:

    ```
    <enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-credentials>
    ```

    For more information, see http://download.oracle.com/docs/cd/E12840_01/wls/docs103/security/thin_client.html#understanding_basic_atn.
12. Start the WebLogic Server by running *DOMAIN_HOME*`/bin/startWebLogic.sh` (where `DOMAIN_HOME` is the domain that you created in Step 4.

    For example:

    OUD_BASE_LOCATION_HOME`/user_projects/domains/base_domain/bin/startWebLogic.sh`
13. Deploy the DSML Gateway WAR file, as described in the following section.

17.6.1.2 Deploying the DSML Gateway WAR File

After configuring the WebLogic Server for the DSML Gateway, you need to deploy the DSML Gateway WAR file.

To deploy the DSML Gateway WAR file, follow these steps:

1. Create a `DSML` directory in the addons directory and change to that directory.

   ```
   $ cd OUD_BASE_LOCATION_HOME/ORACLE_HOME/addons
   $ mkdir DSML
   $ cd DSML
   ```

2. Explode the DSML gateway WAR file.

   ```
   $ jar xvf ../OUD-DSML.zip
   ```

3. Edit the DSML configuration, if required.

   The `WEB-INF/web.xml` file includes initialization parameters that can be used to specify the address (in the `ldap.host` parameter) and port number (in the `ldap.port` parameter) of the directory server to which DSML requests should be forwarded.

   By default, the DSML gateway is configured to communicate with a directory server on the same system, that is, `localhost`) on port `389`. If you must change the host address and port number, edit the `web.xml` file and restart the web container.

4. In a browser window, connect to the WebLogic Administration Console (for example, `http://`*hostname*`:7001/console`), where *hostname* is the host on which WebLogic Server is running.

   Use the administrator user name and password that you established in Step 5 of the preceding procedure.

5. Follow the WebLogic Server Documentation to install a Web application (`http://download.oracle.com/docs/cd/E12840_01/wls/docs103/ConsoleHelp/taskhelp/web_applications/InstallWebApplications.html`).

   - In step 4 of the procedure, provide the path to the exploded application (*OUD_BASE_LOCATION_HOME/ORACLE_HOME*`/addons/DSML`).
   - In step 6 of the procedure, select **Install this deployment as an application**.
   - Accept the default values for the other steps.

6. On the left panel of the Administration Console, click **Deployments**.
7. Select the check box next to the DSML application and click **Start** then **Servicing all requests**.
8. On the Start Deployments panel, click **Yes**.
9. The DSML application is now deployed and available for use.

17.6.2 Confirming the DSML Gateway Deployment

After the DSML gateway has been deployed and configured, you can communicate with it using any DSMLv2 client.

The following sections describe two ways to accomplish this:

- Confirming the DSML Gateway Deployment Using JXplorer
- Confirming the DSML Gateway Deployment Using the Directory Server Resource Kit

17.6.2.1 Confirming the DSML Gateway Deployment Using JXplorer

The JXplorer tool is a Java-based LDAP browser that can be used to browse, search, and edit the contents of an Oracle Unified Directory instance. This tool can communicate using both LDAP and DSML. Although JXplorer's DSML support does not allow authentication (and

therefore is restricted to the set of operations available to anonymous users), it is still possible to use it to verify that the DSML gateway is functioning as expected.

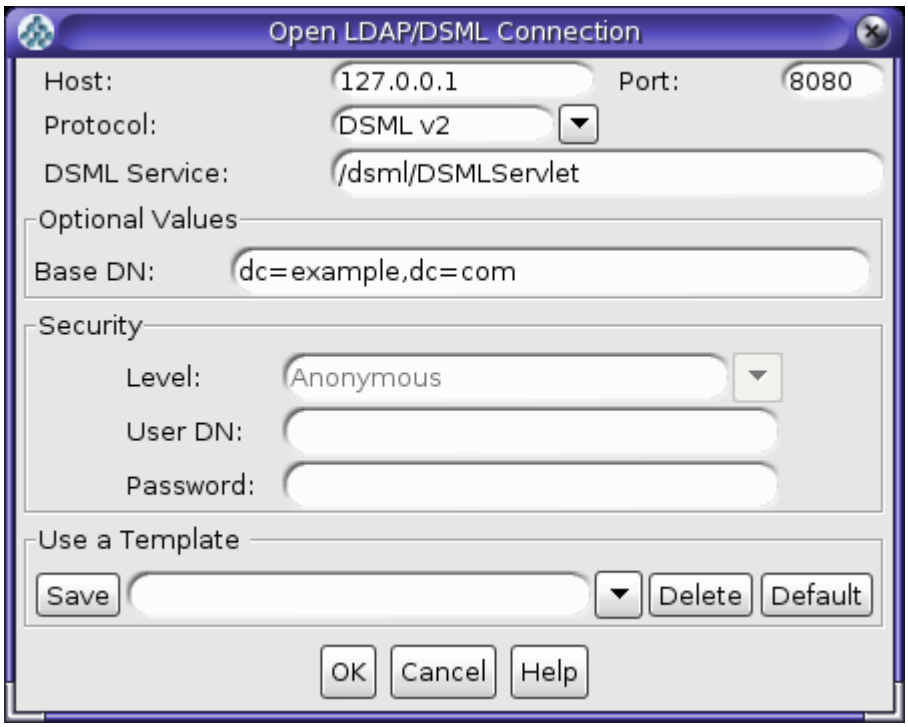
You can download JXplorer, and the accompanying documentation, at jxplorer.org.

To confirm a DSML gateway using JXplorer, follow these steps:

1. Start JXplorer and choose the Connect option from the File menu.

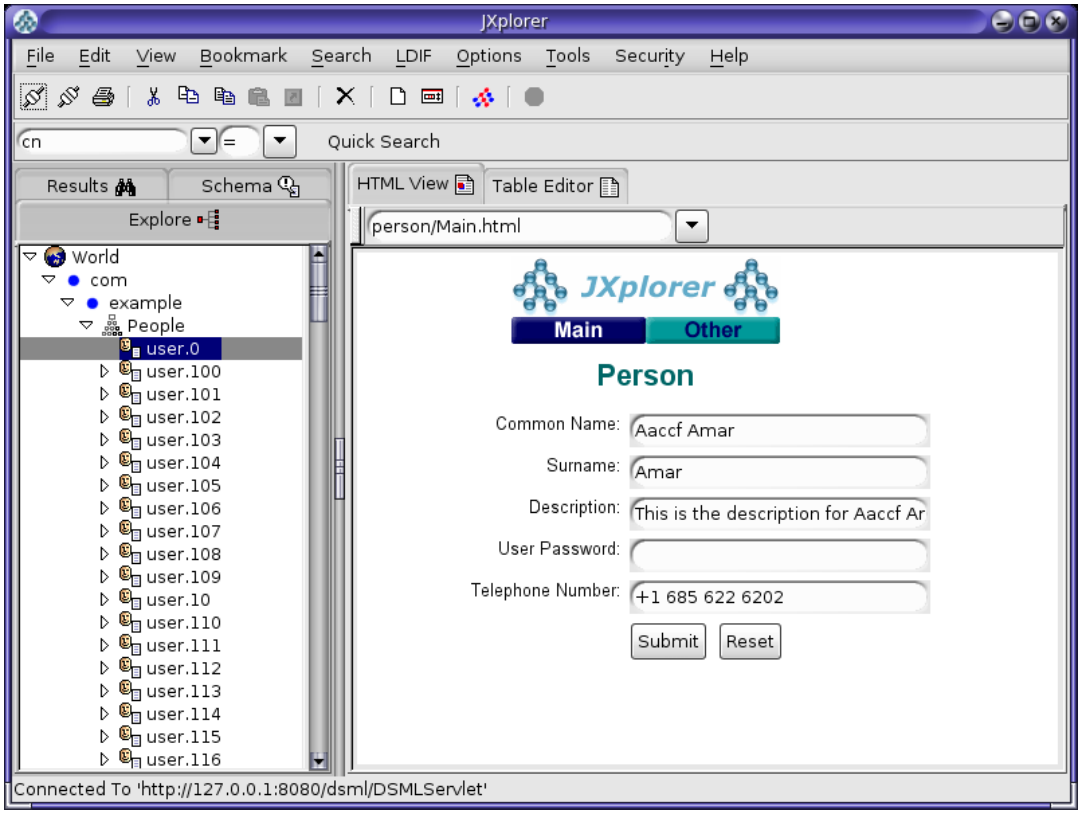
   The Open LDAP/DSML Connection dialog opens with fields for connection information. The following figure shows typical entries.

Figure 17-1 Example Settings for Open LDAP/DSML Connection Dialog



2. Enter the address and port number of the Web application on which the DSML gateway is running.
3. Choose `DSMLv2` from the Protocol list.
4. Specify the path to the DSMLServlet in the DSML Service field.
5. Provide an appropriate base DN value for your directory.
6. Click OK to connect the directory server and display a JXplorer window where you can search and browse the tree (with the limitations imposed for anonymous users).

Figure 17-2 Browsing the JXplorer Tree



17.6.2.2 Confirming the DSML Gateway Deployment Using the Directory Server Resource Kit

The Directory Server Resource Kit (DSRK) is a collection of utilities that can be used with directory servers. The DSRK was originally intended for use with Oracle Directory Server Enterprise Edition, but in most cases the applications also work with Oracle Unified Directory. The most recent version of the DSRK is included as part of Oracle Directory Server Enterprise Edition 11g Release 1 PS1 (11.1.1.7.0), and contains the `dsmlsearch` and `dsmlmodify` tools that can interact with a directory server using DSML rather than LDAP.

> **Note:**
>
> Although an older version of these DSML tools was provided with earlier versions of the Directory Server Resource Kit, the version provided with Oracle Directory Server Enterprise Edition 11g Release 1 PS1 (11.1.1.7.0) is strongly recommended because it is easier to use.
>
> You can download Oracle Directory Server Enterprise Edition 11g Release 1 PS1 (11.1.1.7.0) from the Oracle Software Delivery Cloud site at:
>
> `http://edelivery.oracle.com/`

The following topics explain how to confirm the DSML gateway deployment:

- Using the `dsmlsearch` Command

- Using the `dsmlmodify` Utility

17.6.2.2.1 Using the `dsmlsearch` Command

The `dsmlsearch` command is a DSML-based counterpart to the `ldapsearch` command. `dsmlsearch` operates in a similar manner to `ldapsearch` but there are certain key differences. To see usage information, invoke the command with no arguments, as in the following example:

```
$ ./dsmlsearch
usage: dsmlsearch -h http://host:port -b basedn [options] filter [attributes...]
where:
-h hostURL URL of the directory server
-b basedn  base dn for search
-D binddn  bind dn
-w passwd  bind password (for simple HTTP authentication)
use "-w - " to prompt for a password
-j pwfile  file where password is stored
-s scope   specify the scope of the search
baseObject - For searching only the base entry
singleLevel - For searching only the children
wholeSubtree - For searching the base entry and all childrens
-a deref   specify how aliases are deferenced
neverDerefAliases - Aliases are never dereferenced
derefFindingBaseObj - Dereferenced when finding the base DN
derefAlways - Dereferenced when finding below the base DN
-l seconds specify the maximum number of seconds to wait for the search
-z number  specify the maximum number of entries to return for the search
-f file    specify the name of the file containing the search filter
```

The `dsmlsearch` command differs in usage from `ldapsearch`:

- The `-h` argument is used to provide a URL to use to access the server. It should include the host and port number, as well as the URI for the gateway servlet (for example, `http://127.0.0.1:8080/dsml/DSMLServlet`).
- The `-b` argument is used to specify the search scope, but notice that the values you provide are different (`baseObject`instead of `base`, `singleLevel`instead of `one`, and `wholeSubtree`instead of `sub`).
- The results are output in DSML format, which is not as user-friendly or human-readable as the LDIF output provided by `ldapsearch`.

Following is an example usage of this tool.

> **Note:**
>
> The DSML output does not contain any line breaks, but line breaks were added to this example for readability

```
$ ./dsmlsearch -h http://127.0.0.1:8080/dsml/DSMLServlet \-b "dc=example,dc=com"
-s baseObject \"(objectClass=*)"
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><dsml:batchResponse xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<dsml:searchResponse><dsml:searchResultEntry dn="dc=example,dc=com"><dsml:attr
name="objectClass"><dsml:value>domain</dsml:value><dsml:value>top</dsml:value>
</dsml:attr><dsml:attr name="dc"><dsml:value>example</dsml:value></dsml:attr>
```

```
</dsml:searchResultEntry><dsml:searchResultDone><dsml:resultCode code="0"/>
</dsml:searchResultDone></dsml:searchResponse></dsml:batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

17.6.2.2.2 Using the `dsmlmodify` Utility

The `dsmlmodify` utility is a DSML-based counterpart to the `ldapmodify` command, and it can perform add, delete, modify, and modify DN operations over DSML. To see the usage information for this tool, run it with no arguments, as shown in this example:

```
$ ./dsmlmodify
usage: dsmlmodify -h http://host:port [options] -f file
where:
-h hostURL URL of the directory server
-D binddn  bind dn
-w passwd  bind password (for simple HTTP authentication)
use "-w - " to prompt for a password
-j pwfile  file where password is stored
-f file    specify the name of the file containing
the modifications
```

As with the `dsmlsearch` utility, the `-h` argument specifies a URL, and the output is returned in DSML form. Unlike `ldapmodify`, the `dsmlmodify` tool does not accept the changes through standard input. Changes must be specified in a file, and that file must be in DSML format instead of than LDIF, and the changes cannot contain an outer `batchRequest` wrapper. The following example shows a typical input file.

```
<addRequest dn="uid=test.user,dc=example,dc=com">
<attr name="objectClass">
<value>top</value>
<value>person</value>
<value>organizationalPerson</value>
<value>inetOrgPerson</value>
</attr>
<attr name="uid">
<value>test.user</value>
</attr>
<attr name="givenName">
<value>Test</value>
</attr>
<attr name="sn">
<value>User</value>
</attr>
<attr name="cn">
<value>Test User</value>
</attr>
<attr name="userPassword">
<value>password</value>
</attr>
</addRequest>
<modifyRequest dn="uid=test.user,dc=example,dc=com">
<modification name="description" operation="replace">
<value>This is the new description</value>
</modification>
</modifyRequest>
<modDNRequest dn="uid=test.user,dc=example,dc=com" newrdn="cn=Test User"
 deleteoldrdn="false" newSuperior="ou=People,dc=example,dc=com" />
<delRequest dn="cn=Test User,ou=People,dc=example,dc=com" />
```

The following example shows the output from applying these changes. Line breaks have been added to the output to make it more readable:

```
$ dsmlmodify -h http://127.0.0.1:8080/dsml/DSMLServlet \  -D "cn=Directory
Manager" -j pwd-file -f /tmp/test.dsml
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><dsml:batchResponse xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<dsml:addResponse><dsml:resultCode code="0"/></dsml:addResponse>
<dsml:modifyResponse><dsml:resultCode code="0"/></dsml:modifyResponse>
<dsml:modDNResponse><dsml:resultCode code="0"/></dsml:modDNResponse>
<dsml:delResponse><dsml:resultCode code="0"/><dsml:errorMessage>The number of
entries deleted was 1</dsml:errorMessage></dsml:delResponse></dsml:batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>

$ dsmlmodify -h http://localhost:8080/dsml/DSMLServlet \  -D "cn=directory
manager" -j pwd-file -f /tmp/dsml.ldif
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><batchResponse xmlns="urn:oasis:names:tc:DSML:2:0:core">
<addResponse><resultCode code="0"/></addResponse>
<modifyResponse><resultCode code="0"/></modifyResponse>
<modDNResponse><resultCode code="0"/></modDNResponse>
<delResponse><resultCode code="0"/></delResponse></batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

17.7 Managing the OUDSM Session Timeout

To ensure that OUDSM is more secure, the default session timeout is five minutes. You can change this OUDSM session timeout setting to a different value from the WebLogic Server Administration Console, as follows:

1. In a browser window, connect to the WebLogic Administration Console (for example, `http://`*hostname*`:7001/console`), where *hostname* is the host on which WebLogic Server is running.

   Enter your administrator user name and password.

2. On the left panel of the Administration Console, in the Domain Structure section, click **Deployments**.

   > **Note:**
   >
   > If you are using the WebLogic Administration Console with the domain configuration locking feature enabled, then you must first go to the Change Center, click **Lock and Edit**, and then go to the Domain Structure section and click **Deployments**.

3. When the Summary of Deployments page is displayed, locate and expand **oudsm** in the Deployments table.
4. Under Modules, click **/oudsm**.
5. When the Settings page is displayed, select the Configuration tab.
6. Change the **Session Timeout** value to the preferred number of seconds (for example 600 seconds).
7. Click **Save**.
8. Save your changes to `Plan.xml` by clicking **OK**.

> **Note:**
>
> You must specify the path to the `Plan.xml` file the first time you execute this procedure. Afterward, you can update its path from the Overview tab on the Settings page.

9. Return to the Deployments page and select **oudsm**.
10. Enable the checkbox next to the OUDSM application row, and then click **Update action**.
11. Enable **Redeploy this application using the following deployment files** and provide the `Plan.xml` file that you saved in step 8.
12. Click **Finish**.

> **Note:**
>
> If you are using the WebLogic Administration Console with the domain configuration locking feature enabled, then you must go to the Change Center and click **Activate Changes**.

13. Login to OUDSM, and then login to the Oracle Unified Directory directory server.
14. Allow the Oracle Unified Directory server to run for awhile.

    After the specified amount of time, you should observe that the session has timed out. In addition,

    - If you set a longer interval, then a Session Timeout pop-up is automatically displayed when the session times out.
    - If you set the session timeout value to a short interval (such as two minutes), then the Session Timeout pop-up is *not* automatically displayed when the session times out. However, if you perform an action that requires connecting to the server, then the pop-up is displayed.

> **Note:**
>
> For more information about using the WebLogic Administration Console with the domain configuration locking feature, see "Use the Change Center" and "Enable and disable the domain configuration lock" in the *Oracle WebLogic Server Online Help*.

18
Managing Directory Data

Understand how to import, export, add, modify, remove, and search data in the directory server from the following topics. This section also includes information about how to make searches more efficient by indexing data, how to ensure that entries are unique, and how to use advanced data features such as virtual attributes, and includes the following topics:

- Importing and Exporting Data
- Importing Large Data Sets
- Backing Up, Purging, and Restoring Data
- About Searching Directory Data
- Using Advanced Search Features
- Handling Directory Data
- Indexing Directory Data
- Reducing Stored Data Size
- Configuring Selective Attribute Caching
- Ensuring Attribute Value Uniqueness
- Configuring Virtual Attributes
- Using LDAP Subentries
- Using Collective Attributes
- Configuring Referrals
- Managing Data Using OUDSM

18.1 Importing and Exporting Data

The directory server provides several mechanisms to move data into and out of a specific back end.

The following topics outline the various options and then describes the import and export mechanisms in more detail:

- Populating a Stand-Alone Directory Server With Data
- Importing Data Using `import-ldif`
- Exporting Data Using `export-ldif`
- About Creating MakeLDIF Template Files

> **Note:**
>
> When you import user entries, be aware that Oracle Unified Directory cannot verify that pre-encrypted passwords match the password policy. Pre-encrypted passwords are therefore rejected with the following error:
>
> ```
> LDAP: error code 53 - Pre-encoded passwords are not allowed for the
> password attribute userPassword.
> ```
>
> To allow pre-encrypted passwords when you import user entries using `ldapmodify` or `import-ldif`, change the default password policy by setting the advanced property `allow-pre-encoded-passwords` to `true`. For more information, see Modifying the Default Password Policy.

18.1.1 Populating a Stand-Alone Directory Server With Data

You can populate a stand-alone directory by importing data from an LDAP Data Interchange Format (LDIF) using `setup` utility or `import-ldif` command, copying the binary database from another server, restoring from a previous backup, if any, or by adding entries using the `ldapmodify` command.

To populate a stand-alone directory server with data, use one of the following methods:

- Import the data from an LDAP Data Interchange Format (LDIF) file while you are setting up the server, either by using the `setup` utility in GUI mode or by using the `setup` utility in interactive command-line mode. This is the most convenient method of initializing a stand-alone server or the first server in a replicated topology.
- Start with an empty suffix and add entries by using the `ldapmodify` command, for example:

  ```
  $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
    -a -f /usr/local/add_entry.ldif
  ```

- Import data from an LDIF file, using the `import-ldif` command. For example:

  ```
  $ import-ldif -b dc=example,dc=com -n userRoot -l /var/tmp/Example.ldif
  ```

  This method is much more efficient for the addition of bulk entries. The `import-ldif` command imports data from an LDIF file either by replacing any existing data in the suffix or by appending data to a base DN. Similarly, the `export-ldif` command exports entries from a database to an LDIF file, which can then be imported to another server. Both tools support file compression, SASL extension, and client/server authentication using SSL and startTLS.

- Copy the binary database from another server. This method is also called *binary copy*.

  ```
  $ cp instance-path/db/example.db destination-path/db
  ```

- Restore the database from a backup using the `restore` command, for example:

  ```
  $ restore -d /home/backup/userRoot
  ```

> **Note:**
>
> Performing a binary database copy or restoring a database from a backup requires the source server and the destination server to have the same database remote LDAP structures and indexes.

18.1.2 Importing Data Using `import-ldif`

The `import-ldif` command is used to populate a directory server back end with data read from an LDIF file or with data generated from MakeLDIF Template files. In most cases, `import-ldif` is significantly faster than adding entries using `ldapmodify`.

For information on creating MakeLDIF files, see About Creating MakeLDIF Template Files. The `import-ldif` command supports both LDIF files and compressed files (`.zip`).

Note the following aspects of an import operation:

- A complete import to an entire Oracle Berkeley DB Java Edition (JE) back end will have better performance than a partial import to a branch of the JE back end. All imported LDIF files must use UTF-8 character-set encoding.
- Importing *suffixes* is a resource-intensive operation. If you import LDIF files that include a large number of suffixes, your system might have insufficient heap to complete the import operation. Before importing such LDIF files, you should therefore increase the heap as much as possible. For more information, see Tuning Performance and Importing Large Data Sets.
- You do not need root privileges to import an LDIF file, but you must authenticate as a user with root permissions, such as `cn=Directory Manager`.

The following sections describe how to import data using the `import-ldif` command:

- About import-ldif Operation Modes
- Importing Data in Offline Mode
- Replacing Existing Data During an Offline Import
- Appending Imported Data to Existing Data
- Importing Fractional Files
- Importing Fractional Files Using Filters
- Including or Excluding Attributes During Import
- Importing a Compressed LDIF File
- Recording Rejected or Skipped Entries During Import
- Importing Data From a MakeLDIF Template
- Running an Import in Online Mode
- Scheduling an Import

18.1.2.1 About import-ldif Operation Modes

The `import-ldif` command has two modes of operation: online and offline.

- **Online mode**. In online mode, `import-ldif` contacts a running directory server instance and registers an import task. The command accesses the task back end over SSL through the administration connector. For more information, see Managing Administration Traffic to the Server. Online mode runs automatically when any connection options (such as `--hostname`, `--port`, `--bindDN`, and `--bindPasswordFile`) are specified.

  > **Note:**
  >
  > Even for an online import, the back end is unavailable during the import. In a replicated topology, the overall service remains available through the referral on update feature. For more information, see Understanding Referrals in a Replicated Topology.
  >
  > In general, if you expect to do online imports, you should increase the heap when you start the server. For more information, see Tuning Performance.

- **Offline mode**. When no connection options are specified, the command runs in offline mode. In offline mode, `import-ldif` accesses the database directly rather than through a directory server instance. In this case, the directory server must be stopped.

18.1.2.2 Importing Data in Offline Mode

The following procedure imports a remote LDAP database with new entries specified in an import LDIF file. The command runs in *offline* mode, which requires the server to be shut down prior to import.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Import the LDIF file, as shown in the following example:

   ```
   $ import-ldif -b dc=example,dc=com -n userRoot -l Example.ldif
   ```

   This command specifies the base DN for the branch of the data that should be included in the import (`-b`), the back-end ID into which the data is imported (`-n`), and the LDIF file used for the import (`-l`).

18.1.2.3 Replacing Existing Data During an Offline Import

The following procedure replaces an existing back-end with new entries specified in an import file.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Import the LDIF file, replacing the existing data. For example:

   ```
   $ import-ldif --includeBranch dc=example,dc=com --backendID userRoot \
     --replaceExisting --ldifFile Example.ldif
   ```

18.1.2.4 Appending Imported Data to Existing Data

The following procedure appends the entries in an import file to the existing entries in the back end.

1. Stop the server if it is running.

```
$ stop-ds
```

2. Import the LDIF file, appending the new data to the existing data. For example:

```
$ import-ldif --backendID userRoot --append --ldifFile new.ldif
```

18.1.2.5 Importing Fractional Files

The `import-ldif` command provides options to import a portion of an import file by specifying the base DN to include or exclude during the process.

This example imports all entries below the base DN, `dc=example,dc=com`, and excludes all entries below `ou=People,dc=example,dc=com`.

1. Stop the server if it is running.

```
$ stop-ds
```

2. Import a portion of the LDIF file. For example:

```
$ import-ldif --includeBranch dc=example,dc=com \
  --excludeBranch ou=People,dc=example,dc=com --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

18.1.2.6 Importing Fractional Files Using Filters

The `import-ldif` command provides options to import part of an import file by using filters for data inclusion or exclusion. Ensure that you fully understand how this mechanism works before you use it.

In this example, the contents of an LDIF file are imported, except those entries that match the search filter `l=Auckland` (that is, `location=Auckland`).

The `--includeFilter` option works in a similar manner to `--excludeFilter`, except that it includes all entries that match the search filter during import

1. Stop the server if it is running.

```
$ stop-ds
```

2. Import a portion of the file by using an exclude filter. For example:

```
$ import-ldif --excludeFilter "(l=Auckland)" --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

18.1.2.7 Including or Excluding Attributes During Import

The `import-ldif` command provides options to include and exclude attributes during import by using the `--includeAttribute` and `--excludeAttribute` options, respectively. Ensure that you fully understand how this mechanism works before you use it.

1. Stop the server if it is running.

```
$ stop-ds
```

2. View the entries of the import file before you start the import.

   The directory server provides useful utilities to search, modify, compare, or delete import files without connecting to the server. You can use the `ldifsearch` command to display an entry in your import file. For example, to display the entry for Sam Carter, use the following command:

```
$ ldifsearch -b dc=example,dc=com --ldifFile Example.ldif "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenname: Sam
uid: scarter
cn: Sam Carter
telephonenumber: +1 408 555 4798
sn: Carter
userpassword: sprain
roomnumber: 4612
mail: scarter@example.com
l: Sunnyvale
ou: Accounting
ou: People
facsimiletelephonenumber: +1 408 555 9751
```

In this entry, notice the presence of the `roomnumber` attribute below the `telephonenumber` attribute.

3. Import the file, excluding the `roomnumber` attribute for all entries.

```
$ import-ldif --excludeAttribute "roomnumber" --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

4. Start the server.

```
$ start-ds
```

5. Perform an `ldapsearch` to verify the import.

The following example shows that the `roomnumber` attribute is now absent from Sam Carter's entry.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com --bindDN "cn=Directory Manager" \
  --bindPassword password "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com \
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Sam
uid: scarter
cn: Sam Carter
sn: Carter
telephoneNumber: +1 408 555 4798
ou: Accounting
ou: People
l: Sunnyvale
mail: scarter@example.com
facsimileTelephoneNumber: +1 408 555 9751
```

18.1.2.8 Importing a Compressed LDIF File

The `import-ldif` utility supports compressed LDIF files.

1. Stop the server if it is running.

```
$ stop-ds
```

2. Import the compressed LDIF file.

```
$ import-ldif --includeBranch dc=example,dc=com
  --excludeBranch "ou=People,dc=example,dc=com" --ldifFile Example.ldif \
  --backendID userRoot --replaceExisting --isCompressed
```

18.1.2.9 Recording Rejected or Skipped Entries During Import

The `import-ldif` command provides a means to write to an output file for any entries that are rejected or skipped during the import process. This file enables easy debugging of an LDIF file. Rejected entries occur when the directory server rejects the added entries due to schema violations. Skipped entries occur when entries cannot be placed under the specified base DN.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Import the file, using the `--rejectFile` and `--skipFile` options.

   You can also use the `--overWrite` option to replace any previous items in the two files. Without the option, the directory server appends new rejected and skipped entries to the existing files.

   ```
   $ import-ldif --backendID userRoot --append --ldifFile new.ldif
     --overwrite --rejectFile rejected.ldif --skipFile skipped.ldif
   ```

3. View the contents of the `rejectFile` and `skipFile` to determine which entries were rejected or skipped during the import. For example:

   ```
   $ more rejected.ldif
   # Entry ou=Contractors,dc=example,dc=com read from LDIF starting at line 1
   is not valid because it violates the server's schema configuration:
   Entry ou=Contractors,dc=example,dc=com violates the Directory Server schema
   configuration because it includes attribute changeType which is not allowed.
   changetype: add objectclasses defined in that entry objectclass: top
   objectclass: organizationalUnit ou: Contractors ou: Product Testing
   ou: Product Dev ou: Accounting ...

   $ more skipped.ldif
   # Skipping entry ou=People,dc=example,dc=com because the DN is not one that should
   be
     included based on the include and exclude branches objectclass: top
     objectclass: organizationalunit ou: People
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr ="userpassword ||
     telephonenumber || facsimiletelephonenumber")(version 3.0;acl "Allow self entry
     modification"; allow (write)(userdn = "ldap:///self");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Accounting)")(version 3.0;acl "Accounting Managers Group
     Permissions"; allow (write)
     (groupdn = "ldap:///cn=Accounting Managers,ou=groups,dc=example,dc=com");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Human Resources)")(version 3.0;acl "HR Group
   Permissions";
     allow write)(groupdn = "ldap:///cn=HR Managers,ou=groups,dc=example,dc=com");)
   aci:
     (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn ||sn || uid")
     (targetfilter ="(ou=Product Testing)")(version 3.0;acl "QA Group Permissions";
   allow
     (write)(groupdn = "ldap:///cn=QA Managers,ou=groups,dc=example,dc=com");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Product Development)")(version 3.0;acl "Engineering
   Group
     Permissions"; allow (write)(groupdn =
     "ldap:///cn=PD Managers,ou=groups,dc=example,dc=com");) ...
   ```

18.1.2.10 Importing Data From a MakeLDIF Template

The directory server includes the Java utility, `makeLDIF`, that can be used to generate sample data for import. The `makeLDIF` utility requires a template file. You can create your own template file, or you can use the template file located in `INSTANCE_DIR/OUD/config/MakeLDIF/example.template`, editing it as required. For more information, see About Creating MakeLDIF Template Files and make-ldif.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Import the data, using a template file.

   The sample template generates 10,003 sample entries in the specified back end.

   ```
   $ import-ldif --backendID userRoot --templateFile example.template \
     --randomSeed 0
   ```

18.1.2.11 Running an Import in Online Mode

The `import-ldif` utility can also be run with the server online. In online mode, the command accesses the task back end over SSL through the administration connector. To run the command in online mode you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates. For more information, see Managing Administration Traffic to the Server.

Run the `import-ldif` command with the appropriate connection options.

```
$ import-ldif -h localhost -port 4444 -D "cn=Directory Manager" -j pwd-file -X \
  -l /ldif-files/example.ldif
```

18.1.2.12 Scheduling an Import

The `import-ldif` utility provides a `--start` option for scheduling the import at some future date. You can view this scheduled task by using the `manage-tasks` utility. The command accesses the task back end over SSL through the administration connector. To schedule an import task, you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates.

Run the `import-ldif` command with the `--start` option.

```
$ import-ldif -h localhost -port 4444 -D "cn=Directory Manager" -j pwd-file -X \
  -l /ldif-files/example.ldif --start 20080124121500
```

For more information, see Configuring Commands As Tasks.

18.1.3 Exporting Data Using `export-ldif`

The `export-ldif` command is used to export data from a directory server back end.

The command is useful for the following tasks:

- Backing up directory data
- Exporting data to another application
- Repopulating a database after a change to the directory topology
- Reinitializing master servers in a replicated topology

> **Note:**
>
> The `export-ldif` command cannot be used to export data from the following back ends: `monitor`, `ads-truststore`, `backup`, and `config-file-handler`.

The following sections describe how to export data using the `export-ldif` command:

- About export-ldif Operation Modes
- Exporting Data to LDIF
- Exporting Partial Data
- Exporting Part of a Back End Using Filters
- Including or Excluding Attributes During Export
- Exporting to LDIF and Then Compress the File
- Running an Export in Online Mode
- Scheduling an Export

18.1.3.1 About export-ldif Operation Modes

The `export-ldif` command has two modes of operation: online and offline.

- **Online mode**. In online mode, `export-ldif` contacts a running directory server instance and registers an export task. This mode runs automatically when the LDAP connection options (`--hostname`, `--port`, `--bindDN`, and `--bindPasswordFile`) are used. The command accesses the task back end over SSL through the administration connector. For more information, see Managing Administration Traffic to the Server.
- **Offline mode**. When no connection options are specified, the command runs in offline mode. In offline mode, `export-ldif` accesses the database directly rather than through a directory server instance. In this case, the directory server must be stopped.

For more information, see export-ldif.

18.1.3.2 Exporting Data to LDIF

This procedure explains how to export data to LDIF. Follow these steps:

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Export the back end to a specified LDIF file.

   ```
   $ export-ldif --includeBranch "dc=example,dc=com" --backendID userRoot \
     --ldifFile example.ldif
   ```

18.1.3.3 Exporting Partial Data

The `export-ldif` command provides options to export a part of a back end by specifying the base DN and its children for inclusion or exclusion during processing.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Export a portion of the back end.

   In this example, only the entries under `ou=People,dc=example,dc=com` are exported.

   ```
   $ export-ldif --includeBranch ou=People,dc=example,dc=com --backendID userRoot \
     --ldifFile example-people.ldif
   ```

3. Use the `ldifsearch` command to verify the exported file.

   The `ldifsearch` command verifies entries in an LDIF file without connecting to the directory server. You can use it in a manner similar to the `ldapsearch` command. For example:

   ```
   $ ldifsearch -b dc=example,dc=com --ldifFile export.ldif "(objectclass=*)"
   dn: ou=People,dc=example,dc=com
   objectClass: organizationalunit
   objectClass: top
   ou: People
   dn: uid=scarter,ou=People,dc=example,dc=com
   objectClass: person
   objectClass: organizationalPerson
   objectClass: inetOrgPerson
   objectClass: top
   givenName: Sam
   uid: scarter
   cn: Sam Carter
   sn: Carter
   telephoneNumber: +1 408 555 4798
   userPassword: {SSHA}Ocpp2P4sImz2MziL69AUG9+khdIhFpmU4B5mvA==
   roomNumber: 4612
   ou: Accounting
   ou: People
   l: Sunnyvale
   mail: scarter@example.com
   facsimileTelephoneNumber: +1 408 555 9751 ...
   ```

18.1.3.4 Exporting Part of a Back End Using Filters

The `export-ldif` command provides options to export part of a back end by using a search filter. The directory server includes or excludes all entries that match the filter. Ensure that you fully understand how this mechanism works before you use it.

In this example, only those entries that match the search filter `l=Cupertino` (that is, `location=Cupertino`) are exported. The `--excludeFilter` option works in a similar manner to `--includeFilter`, except that it excludes all entries that match the filter during export.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Export a portion of the back end by using the `--includeFilter` option.

   ```
   $ export-ldif --includeFilter "(l=Cupertino)" --backendID userRoot \
     --ldifFile export.ldif
   ```

18.1.3.5 Including or Excluding Attributes During Export

The `export-ldif` utility provides options to include and exclude attributes during export by using the `--includeAttribute` and `--excludeAttribute` options, respectively. Ensure that you fully understand how this mechanism works before you use it.

1. With the server running, view a sample entry, by using the `ldapsearch` command. For example:

```
$ ldapsearch --baseDN dc=example,dc=com "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenname: Sam
uid: scarter
cn: Sam Carter
telephonenumber: +1 408 555 4798
sn: Carter
userpassword: sprain
roomnumber: 4612
mail: scarter@example.com
l: Sunnyvale
ou: Accounting
ou: People
facsimiletelephonenumber: +1 408 555 9751
```

2. Stop the server.

```
$ stop-ds
```

3. Export the back end, using the `--includeAttribute` option to specify the attributes that should be included in the export.

   You can use the `--includeAttribute` option multiple times for each attribute that should be included. In this example, only the top level attributes are exported.

```
$ export-ldif --backendID userRoot --includeAttribute dn --includeAttribute dc \
  --includeAttibute cn --includeAttribute sn --includeAttribute givenname \
  --includeAttribute objectclass --includeAttribute ou --includeAttribute uid \
  --ldifFile export.ldif
```

4. Use the `ldifsearch` command to verify the export file.

   If an error occurs, the server continues processing the command.

```
$ ldifsearch --baseDN dc=example,dc=com --ldifFile export.ldif "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniquenames
objectClass: top
cn: Directory Administrators
ou: Groups
dn: ou=People,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: People ...
```

18.1.3.6 Exporting to LDIF and Then Compress the File

The `export-ldif` command allows you to compress the output LDIF file.

1. Stop the server if it is running.

   ```
   $ stop-ds
   ```

2. Export to LDIF and then compress the file.

   ```
   $ export-ldif --backendID userRoot --ldifFile export.ldif --compress
   ```

18.1.3.7 Running an Export in Online Mode

The `export-ldif` command can also be run with the server online. In online mode, the command accesses the task back end over SSL through the administration connector. For more information, see Managing Administration Traffic to the Server. To run the command in online mode you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates.

Run the `export-ldif` command with the LDAP connection options. For example:

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot --ldifFile export.ldif
```

18.1.3.8 Scheduling an Export

The `export-ldif` utility provides a `--start` option for scheduling the export at some future date. You can view this scheduled task by using the `manage-tasks` utility. The command accesses the task back end over SSL through the administration connector. For more information, see Managing Administration Traffic to the Server.

To schedule an export task, you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates.

The server must be running to schedule an export.

Run the `export-ldif` command with the `--start` option and the LDAP connection parameters.

The `--start` option takes as its value a date and time in the format `yyyymmddhhmmss`. For example:

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot \
  --ldifFile export.ldif --start 20080124121500
```

18.1.4 About Creating MakeLDIF Template Files

The `make-ldif` command can use template files to define the way in which LDIF files are to be generated. This approach allows for flexibility without the need to alter any code to produce the desired result.

The topics in this section describe how to use the `make-ldif` command to create customized LDIF files:

- Understanding the Template File Format
- Understanding make-ldif Template File Tags

18.1.4.1 Understanding the Template File Format

Template files can contain up to four sections, which must be provided in the following order:

1. Understanding Custom Tag Includes

2. Understanding Global Replacement Variables
3. Understanding Branch Definitions
4. Understanding Template Definitions

18.1.4.1.1 Understanding Custom Tag Includes

Custom tag includes provide a mechanism for loading custom tags and making them available for use when processing `make-ldif` templates. This should be done using the `include` directive, as follows:

```
include com.example.opends.makeldif.MyCustomTag
```

The specified class must be in the class path, and it must be a subclass of the `org.opends.server.tools.makeldif.Tag` class. For information about developing custom tags, see Defining Custom Tags.

All of the standard replacement tags that are provided with `make-ldif` are automatically available for use and therefore do not require an explicit `include` directive.

18.1.4.1.2 Understanding Global Replacement Variables

The first section that should be present in the template file is the section that defines the global replacement variables. Global replacement variables are used to define strings of text that can be referenced later in the template file and are automatically replaced as each line is read into memory (much like a C preprocessor replaces macros in code with their defined values). For example, the following replacement variable definition creates a global replacement variable named `suffix` with a value of `dc=example,dc=com`:

```
define suffix=dc=example,dc=com
```

When a global replacement variable is defined, any case in which that variable name appears in square brackets (for example, `[suffix]`), causes the token to be replaced with the value that has been defined for that replacement variable.

When all the replacement variable definitions have been read (as signified by the first blank line following one or more replacement variable definitions), all remaining lines that are read from the template file are processed on a line-by-line basis. Any occurrences of a replacement variable name in square brackets are replaced with the value of that variable. Because that replacement is done as the template file is read into memory, replacement variables can occur in any point, including branch and template definitions, and even inside tags.

If there are global replacement variables defined in the template file, they must appear at the top of the file and there should not be any spaces between them. However, replacement variables are not required. If there are no replacement variables, the template file must start with the branch definitions.

18.1.4.1.3 Understanding Branch Definitions

Branch definitions are used in make-ldif template files to define the basic structure to use for the generated LDIF. They specify the entry or entries that should appear at the top of the hierarchy, and the number and types of entries that should appear below them.

The most basic form of a branch definition is as follows:

```
branch: dc=example,dc=com
```

This example specifies that the following entry is to be created with a DN of `dc=example,dc=com`:

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
dc: example
```

The basic structure of the entry is defined by the RDN attribute of `dc` specified in the DN of the branch definition. The `make-ldif` command automatically associates the `dc` RDN attribute with the `domain` object class. The `make-ldif` command has similar definitions for other common RDN attributes in branch entries:

o
Creates an entry with the `organization` object class.

ou
Creates an entry with the `organizationalUnit` object class.

c
Creates an entry with the `country` object class.

You can also use any other kind of RDN attribute for a branch entry. For branch entries with an RDN attribute other than the ones specified above, the entry is created with the `untypedObject` and `extensibleObject` object classes.

The branch definition provided above does not cause any additional entries to be created below that branch entry. To do this, you must specify one or more `subordinateTemplate` lines. For example:

```
branch: ou=People,dc=example,dc=com
subordinateTemplate: person:100
```

This causes the `ou=People,dc=example,dc=com` entry to be created, and then 1000 other entries created below it modeled after the `person` template. The `person` template should be defined later in the template file. For more information, see Understanding Template Definitions.

Branch entries are not limited to just one `subordinateTemplate` definition. You can specify multiple `subordinateTemplate` definitions by including them on separate lines of the branch definition. The following example creates 1000 entries based on the `person` template and an additional 100 entries based on the `certificatePerson` template:

```
branch: ou=People,dc=example,dc=com
subordinateTemplate: person:10000
subordinateTemplate: certificatePerson:100
```

In all of the examples described previously, the branch entries themselves contain only the DN, the RDN attribute, and the object classes associated with the RDN attribute. You can include any other attributes in the branch entry by including them in the branch definition in the template file. For example, the branch definition:

```
branch: dc=example,dc=com
description: This is the description for dc=example,dc=com
```

creates the entry:

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
```

```
dc: example
description: This is the description for dc=example,dc=com
```

This additional text can be static, can contain any defined global replacement variables, or can contain a subset of the replacement tags that can be used in template definitions. For an overview of the tags available and information about which tags can be used in branch definitions, see Understanding Standard Replacement Tags.

18.1.4.1.4 Understanding Template Definitions

The heart of the `make-ldif` template file structure is the set of template definitions. Templates define the structure of the entries that are generated. They specify the set of attributes that should be included in the entries and the types of values that those attributes should contain. The specification of values is handled through tags that are parsed by `make-ldif` and replaced with the appropriate values for those tags.

A sample template definition might look as follows:

```
template: person
rdnAttr: uid
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
givenName: <first>
sn: <last>
cn: {givenName} {sn}
initials: {givenName:1}<random:chars:ABCDEFGHIJKLMNOPQRSTUVWXYZ:1>{sn:1}
employeeNumber: <sequential:0>
uid: user.{employeeNumber}
mail: {uid}@[maildomain]
userPassword: password
telephoneNumber: <random:telephone>
homePhone: <random:telephone>
pager: <random:telephone>
mobile: <random:telephone>
street: <random:numeric:5> <file:streets> Street
l: <file:cities>
st: <file:states>
postalCode: <random:numeric:5>
postalAddress: {cn}${street}${l}, {st}  {postalCode}
description: This is the description for {cn}.
```

This example illustrates how `make-ldif` provides some flexibility when generating LDIF data. The tags that can be included in a template definition are described in the topics that follow (see Understanding Standard Replacement Tags and Using Attribute Value Reference Tags).

At the top of the template definition are two lines that provide information about the template itself and are not included in the entries created from this template. The first line specifies the name of the template. This is the name that is referenced in the `subordinateTemplate` lines of the branch definition. The second line specifies the name of the attribute that should be used as the RDN attribute for the entry. The RDN attribute must be assigned a value in the body of the template definition, and the way in which the value is assigned must ensure that the value will be unique among all other entries created with the same template below the same parent.

> **Note:**
>
> It is possible to specify multi-valued RDNs by separating the attribute names with a plus sign, as shown in the following example:
>
> ```
> rdnAttr: uid+employeeNumber
> ```

If multi-valued RDNs are used, all of the RDN attributes must be defined values in the template body and the combination of the RDN values for each entry must be unique. However, it is possible for one or more of the attributes in the RDN to be non-unique if the combination is never duplicated.

In addition to the `template` and `rdnAttr` lines, you can include one or more `subordinateTemplate` lines, which enables you to include dynamically-generated entries below other dynamically generated entries (for example, if each user entry has one or more entries below it), and to allow for complex hierarchies. Although there is no limit placed on this level of nesting, you must ensure that no recursive loops are created by having a `subordinateTemplate` that either directly or indirectly will create additional entries using the same template.

Template definitions also support the concept of inheritance with the `extends` keyword. For example, entries generated from the following template definition include all of the attributes defined in the `person` template as well as `userCertificate;binary` with the specified format:

```
template: certificatePerson
rdnAttr: uid
extends: person
userCertificate;binary:: <random:base64:1000>
```

Multiple inheritance is allowed (by including multiple lines with the `extends` keyword), but as with the `subordinateTemplate` keyword it is important not to create a recursive loop in which a template file could either directly or indirectly inherit from itself.

18.1.4.2 Understanding make-ldif Template File Tags

To ensure that `make-ldif` can generate LDIF files that can be used to simulate a wide variety of deployments, a large number of tags have been defined for use in templates. You can also create custom tags, as described in Defining Custom Tags.

The following topics describe the standard set of tags that can be used in a `make-ldif` template file:

- Understanding Standard Replacement Tags
- Using Attribute Value Reference Tags
- About Tag Evaluation Order

18.1.4.2.1 Understanding Standard Replacement Tags

The `make-ldif` standard replacement tags are special elements that are enclosed in angle brackets (beginning with a less-than sign (`<`) and ending with a greater-than sign (`>`) that are dynamically replaced with generated values. Some standard replacement tags do not require any arguments (for example, `<first>`). Others do take arguments, in which case the tag name comes first followed by a colon and the argument list with a colon between each argument (for

example, <random:numeric:5>). The tag name is treated in a case-insensitive manner, although the arguments are generally case sensitive.

The following types of standard replacement tags are currently included as part of make-ldif:

- The DN Tag

  The DN standard replacement tag is replaced with the DN of the current entry. If that DN is not yet available (for example, because the RDN attribute has not yet been assigned a value in the entry being generated), it is replaced with an empty string. In general, you should ensure that all RDN attributes are assigned values earlier in the template before this tag is used.

  The DN tag can be used without any arguments (for example, <DN>), in which case it is replaced with the full DN of the entry. The tag can also take a single integer argument, which specifies the maximum number of components to include in the output. For example, the tag <DN:1> will only include the left most DN component (often called the RDN) for the entry. So if the entry being generated will have a DN of uid=john.doe,ou=People,dc=example,dc=com, the tag <DN:1> will be replaced with uid=john.doe. If the argument value is negative rather than positive, then it takes the absolute value of the given argument value and takes that number of components from the end of the DN. For example, using a DN of uid=john.doe,ou=People,dc=example,dc=com the tag <DN:-1> is replaced with dc=com.

  This tag can be used in both branch and template definitions.

- The File Tag

  The File standard replacement tag is replaced with a line from a specified file. It requires either one or two arguments. The first argument is the path to the data file, and can be either an absolute path or the name of a file (with no path information) that is contained in the config/MakeLDIF directory. If there is a second argument, it must have a value of either sequential or random, which indicates whether the lines in the file should be taken in sequential order or chosen at random. If the second argument is not provided, the values are selected at random. For example, the tags <file:cities> and <file:cities:random> both cause the tag to be replaced with a randomly-selected line from the cities file, but the tag <file:cities:sequential> causes the city names to be taken in sequential order. If sequential ordering is used and all values are exhausted, it will wrap back around to the first line of the file.

  The make-ldif command includes several standard data files that can be used in generated data. These files are included in the config/MakeLDIF directory and therefore only the filename is required. The files include:

  cities — contains a list of common city names

  first.names — contains a list of common first names

  last.names — contains a list of common last names

  states — contains a list of all two-character US state abbreviations

  streets — contains a list of common street names

  This tag can be used in both branch and template definitions.

- The First Tag

  The First standard replacement tag is replaced with a first name taken from the config/MakeLDIF/first.names file. There is a special relationship between the <first> and <last> tags such that the combination of the first and last names is always unique. When every possible combination from the first and last name files has been exhausted, make-

ldif appends an integer value onto the last name to ensure that the value always remains unique.

The `<first>` tag does not take any arguments. It can be used only in template definitions. It is not allowed for use in branch definitions.

- The GUID Tag

  The GUID standard replacement tag is replaced with a randomly generated GUID (globally-unique identifier) value. All GUID values generated are guaranteed to be unique. The values generated consist of 32 hexadecimal digits in dash-delimited groups of 8, 4, 4, 4, and 12 digits, respectively (for example, `12345678-90ab-cdef-1234-567890abcdef`).

  The `<guid>` tag does not take any arguments. It can be used in both branch and template definitions.

- The IfAbsent Tag

  The IfAbsent standard replacement tag does not generate any value of its own, and is therefore always be replaced with an empty string. However, its value is that it can prevent an attribute from appearing in the entry altogether based on whether a specified attribute or attribute value exists.

  For example, consider the following template:

  ```
  template: example
  rdnAttr: cn
  objectClass: top
  objectClass: untypedObject
  objectClass: extensibleObject
  cn: <guid>
  displayName: <presence:50>{cn}
  description: <ifabsent:displayName>{cn}
  ```

  In this case, the `description` attribute is only included in the generated entry if the `displayName` attribute is not included (that is, the resulting entry will contain either `displayName` or `description` but not both).

  The `IfAbsent` tag requires either one or two arguments. The first argument is the name of the target attribute. If there is a second argument, it specifies a particular value for the target attribute. If a value is provided, the `IfAbsent` tag takes action if that value is included in the generated entry.

  This tag can be used in both branch and template definitions.

- The IfPresent Tag

  The `IfPresent` standard replacement tag does not generate any value of its own, and is therefore always replaced with an empty string. However, its value is that it can prevent an attribute from appearing in the entry altogether based on whether a specified attribute or attribute value exists.

  For example, consider the following template:

  ```
  template: example
  rdnAttr: cn
  objectClass: top
  objectClass: untypedObject
  objectClass: extensibleObject
  cn: <guid>
  displayName: <presence:50>{cn}
  description: <ifpresent:displayName>{cn}
  ```

In this case, the `description` attribute will only be included in the generated entry if the `displayName` attribute is also included (that is, the resulting entry will either contain neither attribute or it will contain both attributes).

The `IfPresent` tag requires either one or two arguments. The first argument is the name of the target attribute. If there is a second argument, it specifies a particular value for the target attribute. If a value is provided, the IfPresent tag will only act if that value is included in the generated entry.

This tag can be used in both branch and template definitions.

- The Last Tag

  The `Last` standard replacement tag is replaced with a last name taken from the `config/MakeLDIF/last.names` file. There is a special relationship between the `<first>` and `<last>` tags such that the combination of the first and last names will always be unique. When every possible combination from the first and last name file has been exhausted, `make-ldif` will append an integer value onto the last name to ensure that the value always remains unique.

  The `<last>` tag does not take any arguments. It can only be used in template definitions. It is not allowed for use in branch definitions.

- The List Tag

  The List standard replacement tag is replaced with a string selected from a provided list of values. The values to use should be provided as arguments to the List tag (at least one argument must be provided). Optionally, each value can be followed with a semicolon and an integer value that specifies the relative weight for that value. If a value does not include a weight, the weight for that item is assumed to be one. The weight is used to control how frequently the associated value is chosen compared with all of the other values in the list.

  For example, to select from a list of the colors red, green, and blue in which all listed colors have equal weights, you can use:

  ```
  <list:red:green:blue>
  ```

  If the color red is to appear twice as frequently as either of the other colors, you can use:

  ```
  <list:red;2:green;1:blue;1>
  ```

  In this case, the `1` following the green and blue elements are not technically needed because the weight of any item that does not explicitly include a weight is one, but it is provided in the example above for clarity.

  This tag can be used in both branch and template definitions.

- The ParentDN Tag

  The `ParentDN` standard replacement tag is replaced with the DN of the parent entry of the entry being generated. This should always be available.

  This tag does not take any arguments. It can only be used in template definitions. It cannot be used in branch definitions.

- The Presence Tag

  The `Presence` standard replacement tag does not generate any value of its own, and is therefore always replaced with an empty string. However, its value is that it can be used to cause the associated attribute to appear in the entry a specified percentage of the time.

  For example, consider the following template:

  ```
  template: example
  rdnAttr: cn
  ```

```
objectClass: top
objectClass: untypedObject
objectClass: extensibleObject
cn: <guid>
displayName: <presence:50>{cn}
```

In this case, the `displayName` attribute will only be present in about 50% of the entries generated.

The `Presence` tag requires exactly one argument, which is an integer value between 0 and 100, indicating the percentage of entries that should have the associated attribute.

This tag can be used in both branch and template definitions.

- The Random Tag

  The `Random` standard replacement tag is replaced with a randomly-generated value. Many different types of values can be generated. This tag accepts a variable number of arguments, but the first argument always specifies the type of value to generate. That type may be one of the following values:

  - **alpha.** This value causes the tag to be replaced with a specified number of lowercase ASCII alphabetic characters (that is, the character set `abcdefghijklmnopqrstuvwxyz`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:alpha:5>` generates a string of five randomly-selected alphabetic characters.
  - **numeric.** This causes the tag to be replaced with one or more numeric digits. There can be either one or two additional arguments. If there is one additional argument, it specifies the number of numeric digits to include in the value (for example, `<random:numeric:5>` will generate a string of five numeric digits). If there are two additional arguments, they will specify the upper and lower bounds for a randomly-generated number (for example, `<random:numeric:5:10>` will generate a random integer between 5 and 10, inclusive).
  - **alphanumeric.** This causes the tag to be replaced with a specified number of lowercase ASCII alphabetic characters (that is, the character set `abcdefghijklmnopqrstuvwxyz`), numeric digits (that is, the character set `0123456789`), or both. This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:alphanumeric:5>` will generate a string of five randomly-selected alphanumeric characters.
  - **chars.** This causes the tag to be replaced with characters from a user-defined character set. This can take either two or three additional arguments. The first additional argument is the characters for the user-defined character set. If there is a single argument after the character set, it specifies the number of characters to take from that set (for example, `<random:chars:abcd:3>` will cause three characters to be chosen in which each of those characters is either a, b, c, or d). If there are two arguments after the character set, they must be integer values and the number of characters generated will be an integer between this range (for example, `<random:chars:abcd:3:5>` will cause between 3 and 5 characters to be included in the value, where each character is either a, b, c, or d).
  - **hex.** This causes the tag to be replaced with a specified number of hexadecimal characters (that is, the character set `0123456789abcdef`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:hex:5>` will generate a string of five randomly-selected hexadecimal characters.

- **base64.** This causes the tag to be replaced with a specified number of characters allowed in the base64 character set (`ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01234567890+/`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:base64:5>` will generate a string of five randomly-selected hexadecimal characters.
- **month.**This causes the tag to be replaced with the name of a month of the year. If there are no additional arguments, the full name of the month is included (for example, `<random:month>` might return a value of `October`). If there is a single additional argument, it must be an integer value that specifies the maximum number of characters to include from the name of the month (for example, `<random:month:3>` might generate a value of `Oct`).
- **telephone.**This causes the tag to be replaced with a randomly-generated telephone number in the format 123-456-7890. It does not take any additional arguments (that is, it should always be used like `<random:telephone>`).

This tag can be used in both branch and template definitions.

- The RDN Tag

  The `RDN` standard replacement tag is replaced with the RDN (that is, the leftmost DN component) of the current entry. If the RDN is not yet available (for example, because the RDN attribute has not yet been assigned a value in the entry being generated), it will be replaced with an empty string. In general, you should ensure that all RDN attributes are assigned values earlier in the template before this tag is used. The behavior of this tag is identical to that of the DN tag when used with a single argument whose value is one (that is, `<dn:1>`).

  The `RDN` tag does not take any arguments. It can be used in both branch and template definitions.

- The Sequential Tag

  The `Sequential` standard replacement tag is replaced with an integer value. Each entry is given a sequentially-incrementing value (for example, the first entry is given a value of zero, the next entry a value of one, and so on).

  This tag can take zero, one, or two arguments:

  - If there are no arguments (that is, the tag is `<sequential>`), the first value will be zero, and the value will be reset to zero for each new branch.
  - If there is a single argument, it must be an integer that specifies the initial value to use (for example, a tag of `<sequential:1000>` will start generating values at 1000 instead of 0). The value will be reset to the specified initial value for each new branch.
  - If there are two arguments, the first must be an integer that specifies the initial value, and the second should be a Boolean value of either `true` or `false` indicating whether to reset the counter each time a new branch is started.

  This tag can be used in both branch and template definitions.

- The _DN Tag

  The `_DN` (note the leading underscore character) standard replacement tag is replaced with the DN of the entry being generated, but with an underscore used instead of a comma between DN components. Apart from using underscores instead of commas, this works exactly like the DN tag. As such, it can also take an optional integer argument that specifies the number of components from the left (or from the right if the value is negative) should be included.

This tag can be used in both branch and template definitions.

- The _ParentDN Tag

  The `_ParentDN` (note the leading underscore character) standard replacement tag is replaced with the DN of the parent entry of the entry being generated, but with an underscore used instead of a comma between DN components. This should always be available.

  This tag does not take any arguments. It can only be used in template definitions. It cannot be used in branch definitions.

18.1.4.2.2 Using Attribute Value Reference Tags

Attribute value reference tags can be used to replace the tag with the value of a specified attribute from the same entry. They are used by enclosing the name of the desired attribute in curly braces. For example, `{cn}` will be replaced with the value of the `cn` attribute, if it has already been given a value in the target entry. If the target attribute has not yet been given a value in the entry, the tag will be replaced with an empty string.

For example, consider the following excerpt from a template:

```
givenName: <first>
sn: <last>
uid: {givenName}.{sn}
cn: {givenName} {sn}
mail: {uid}@example.com
```

If the value chosen for the first name is `John` and the last name is `Doe`, then the resulting LDIF output would be:

```
givenName: John
sn: Doe
uid: John.Doe
cn: John Doe
mail: John.Doe@example.com
```

It is also possible to place a colon after the name of the attribute followed by a positive integer value specifying the maximum number of characters to include from the target attribute. For example, the template excerpt:

```
givenName: <first>
sn: <last>
initials: {givenName:1}{sn:1}
```

would cause the following LDIF to be generated:

```
givenName: John
sn: Doe
initials: JD
```

If the specified length is longer than the value of the named attribute, the entire value is used with no padding added. Otherwise, the specified number of characters are taken from the value.

18.1.4.2.3 About Tag Evaluation Order

All tags in the `make-ldif` syntax are currently given equal priority. As such, they are evaluated in the order that they appear in the template definition, from top to bottom, and from left to right within a given line. It is not possible to embed one tag within another.

18.1.4.3 Defining Custom Tags

The make-ldif utility has been designed in an extensible manner so that new tags can be defined and used in template files.

All tags must be subclasses of the `org.opends.server.tools.makeldif.Tag` abstract class.

All of the tags available in `make-ldif` are included in the `org.opends.server.tools.makeldif` package. They may be used for reference to understand what is involved in implementing a custom tag.

> **Note:**
>
> If you define a custom tag, ensure that it is available for use in any template file that might need it. This is done using the `include` statement, that should appear at the top of the template file. For more information, see Understanding Custom Tag Includes.

Custom tag definitions must include the following methods:

Methods	Description
public String getName()	This retrieves the name that should be used to reference the tag. The value that it returns must be unique among all other tags in use by the server.
public boolean allowedInBranch()	This indicates whether the tag will be allowed in branch definitions. If it returns a value of `true`, then the tag may be used in both branch and template definitions. If it returns a value of `false`, then the tag may be used in template definitions but not branch definitions.
public void initializeForBranch(TemplateFile templateFile, Branch branch, String[] arguments, int lineNumber, List<String> warnings)	This performs any initialization that may be required if the tag is to be used in a branch definition. This does not need to be implemented if `allowedInBranch()` returns `false`.
public void initializeForTemplate(TemplateFile templateFile, Template template, String[] arguments, int lineNumber, List<String> warnings)	This performs any initialization that may be required of the tag is to be used in a template definition.
public void initializeForParent(TemplateEntry parentEntry)	This performs any initialization that may be required before starting to generate entries below a new parent. This does not need to be implemented if no special initialization is required.
public TagResult generateValue(TemplateEntry templateEntry, TemplateValue templateValue)	This generates the value that will be used to replace the associated tag when generating entries.

18.2 Importing Large Data Sets

Understand on improving performance when importing large data sets to the directory server from this topic.. By default, the server imports data with a fixed set of parameters.

You can change the default behavior in two ways:

- Specify certain options when you run the `import-ldif` command.

  For more information, see About the Import Options Setup.

- Use the `dsjavaproperties` command to set the appropriate Java arguments before running the `import-ldif` command.

  For more information, see Tuning the JVM and Java Arguments.

18.2.1 About the Import Options Setup

There are various options of `import-ldif` command that are useful when importing particularly large databases. The options are listed out and described in this topic.

They are listed below: following options of the `import-ldif` command are useful when you are importing particularly large databases:

- `--skipDNValidation`

  This option significantly speeds up a large import because no DN validation or database loading is performed during the first phase of the import. The DNs in the LDIF file are treated as regular indexes and are written to a scratch index file that is loaded in phase two of the import.

  During the second phase of the import, limited DN parental checking is performed. During this evaluation, the DNs in the LDIF file are examined to ensure that each DN has a correct parent DN. When a DN is detected without a parent, a dummy entry is written to the reject file.

  If the `--skipDNValidation` option is specified, no duplicate DN checking is performed.

  The server does not remove bad entry IDs from the index database during phase two of the import. It is therefore essential that the LDIF import file is correct if the `--skipDNValidation` option is specified. Generally, you can generate correct LDIF files by using the `make-ldif` command, LDIF files exported from an LDAP server, or LDIF files created by scripts that are historically known to generate correct LDIF files.

- `--threadCount`

  This option speeds up a large import by enabling you to specify that more threads are dedicated to the import process. By default, two threads per CPU are used for an import operation.

  Increasing the `--thread-count` also increases the buffer space that is required in phase one of the LDIF import.

- `--tmpDirectory`

  In the first phase of the import, the server parses the LDIF file, sorts the index records, and writes the records to temporary files. By default, the temporary index files are written to `intall-dir/import-tmp`. If you are importing particularly large index files, you might want to specify another location that has more disk space.

The amount of space required for the temporary index files depends on the following factors:

- The number of entries in the LDIF file.
- The size of the entries in the LDIF file.

  Entries with large numbers of attributes that require indexing will require more space in the temporary directory location, and in the database directory.
- The number of indexes that are configured.

  The more indexes that are configured, the more disk space is required in the temporary directory location, and in the database directory. Substring indexes require more temporary disk space to process than other types of indexes.
- Increasing the `index-entry-limit` for all indexes, or for individual indexes, requires more disk space.

  This is especially true for substring indexes. If you are importing an LDIF file with a large number of entries, turn off all substring indexing to prevent most of the index records from hitting the `index-entry-limit`.

18.2.2 Tuning the JVM and Java Arguments

Tuning the JVM heap is essential to the performance of the `import-ldif` command. Although the `import-ldif` command attempts to limit the amount of JVM heap that it requires, you should allocate as large a JVM heap as possible to `import-ldif` if you are importing a large number of entries.

This section contains the following topics:

18.2.2.1 Considerations for Tuning JVM Arguments

The following JVM tuning considerations have specific impact on the `import-ldif` operation:

- Performing an online import uses the JVM settings that were specified when the server was started. If you plan to import a large LDIF file by using the online import, you should provide extra JVM heap when the server is started. In general, if you must import a large LDIF file, the best option is to perform an offline import.
- The 32-bit JVM generally performs better for smaller LDIF files and for most larger LDIF files.

  You should always try this JVM first, with as large a heap as can be spared. A minimum heap of 2 Gbytes is recommended.
- You might require a 64-bit JVM with a large JVM heap (greater than 4 Gbytes) for extremely large LDIF files, depending on the size of the entries and the indexes configured.

  The 64-bit JVM does not generally perform as well as the 32-bit JVM.
- The default JVM ergonomics might be too small for some JVMs and can seriously impact performance.

  Take note of the default ergonomic values for your JVM (these values differ by vendor and by operating system).

- If you are using replication, you should budget additional JVM heap, particularly if you plan to do a full initialization of the other replicas in the topology after an online import.
- Enable parallel garbage collection for large imports.
- Use the Concurrent Mark Sweep (CMS) garbage collector. This option allows the JVM to minimize the response time of LDAP operations, but it can have a small impact on the overall performance (throughput) of the server.

18.2.2.2 Tuning JVM Arguments

When you have calculated the memory requirement, perform the following steps:

1. Edit the `instance-dir/OUD/config/java.properties` file and set the following values:

```
overwrite-env-java-args=true
import-ldif.offline.java-args=-Xms2560M -Xmx2560M -XX:+UseParallelGC -
XX:+UseConcMarkSweepGC
```

2. Run the `dsjavaproperties` command:

```
$ bin/dsjavaproperties
```

> **Note:**
>
> Running the `dsjavaproperties` command, or setting the `OPENDS_JAVA_ARGS` environment variable, only has a performance impact if the import is offline. If the server is already running and you perform an online import, changing the Java arguments has no impact on the import performance because the import is performed by the server JVM.

18.3 Backing Up, Purging, and Restoring Data

Oracle Unified Directory provides an extensible framework that supports a variety of repository types. The directory server uses the Berkeley DB Java Edition (JE) as its primary back end. The JE back end provides some advantages over other databases as it provides a high-performance, scalable transactional B-tree database with full support for ACID semantics for small to very large data sets. It can also store its entries in encoded form and provide indexes for fast, efficient data retrieval.

This section covers the following topics:

18.3.1 Overview of the Backup and Restore Process

To maintain the directory data on the JE back end, Oracle Unified Directory provides efficient backup and restore utilities that support full and incremental backups. A *full backup* saves the directory data files in the environment as a compressed archive file. An *incremental backup* saves and compresses just those files that have been written since the previous backup, together with a list of names of files that are unchanged since the previous backup. Oracle Unified Directory stores its backup information in a *backup back end* for easy restores.

Directory server backups can be made on the local disks or on remote disks, for example, on network-attached storage (NAS). If you run a backup locally, you should then copy and store the backup on a different machine or file system for security purposes.

Before you start backing up and restoring data, consider the following:

- You must design a workable backup and restore strategy for your directory services system. For example, you can run an incremental backup daily and perform a full backup at least once a week. Test your backup process and your ability to restore regularly. For data restores, many companies restore a directory server from a replicated server, which ensures that the most update copy of the directory data is used. Backup tapes are still needed if the directory data is damaged (for example, missing entries) and the corrupted data has been replicated to other servers.
- Ensure that you have a disaster recovery plan in place. Disaster recovery is necessary when catastrophic events, data corruption, or data tampering occurs. Companies devise their own plans or out source the work to third party specialists. See Backing Up the Directory Server for Disaster Recovery for more information.
- Ensure that you have a place to store your back ups. Store the archived data, configuration directory, schema subdirectory, and installation directory used for your server together in a single location. All these items are required when you restore the server.

18.3.2 Backing Up Data

The directory server provides an efficient command-line utility (`backup`) to back up databases. The `backup` command can be run immediately or scheduled as a task. If the backup is scheduled, the command contacts the server over SSL, using the administration connector, and registers a backup task. If no connection options are specified, the command runs immediately.

The following procedures show the use of the `backup` command in various backup scenarios:

- Backing Up All Back Ends
- Backing Up All Back Ends with Encryption and Signed Hashes
- Performing an Incremental Backup on All Back Ends
- Backing Up a Specific Back End
- Performing an Incremental Backup on a Specific Back End
- Scheduling a Backup as a Task

18.3.2.1 Backing Up All Back Ends

You can back up the back ends by using the `--backUpAll` option.

You must note that when you execute the `backup` command with `--backUpAll` option, the following runtime data is not backed up. You can back up all the other backends.

```
adminRoot:cn=admin data
ads-truststore:cn=ads-truststore
backup:cn=backups
monitor:cn=monitor
```

The following command is run on a standalone directory server and specifies that all databases should be backed up, compresses the backup file, and saves the file to a specified location.

```
$ backup --backUpAll --compress --backupDirectory /tmp/backup
```

The backup directory contains subdirectories for each back end:

```
$ ls /tmp/backup
./ ../ schema/ tasks/ userRoot/
```

The `backup` utility writes the backup to the specified directory and creates a `backup.info` file that provides details about the backup. The directory server assigns a backup ID based on the current date and time. To create your own ID, use the `--backupID` option:

```
$ ls /tmp/backup/userRoot
./ backup.info
../ backup-userRoot-20231003090628Z
```

The `backup.info` file contains detailed information about the current backup.

```
$ more /tmp/backup/userRoot/backup.info

backend_id=userRoot
backup_id=20231003090628Z
backup_date=20231003090629Z
incremental=false
compressed=true
encrypted=false
property.archive_file=backup-userRoot-20231003090628Z
property.last_logfile_size=58339
property.last_logfile_name=00000000.jdb
```

18.3.2.2 Backing Up All Back Ends with Encryption and Signed Hashes

The `backup` utility provides encryption and signed hash support for secure backups. The use of the encryption and signed hash options requires a connection to an online server instance, so the appropriate connection options must be specified.

Run the backup command.

The following command backs up all back ends, compresses them, generates a hash, signs the hash, and encrypts the data.

```
$ backup -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --backUpAll \
  -X --compress --hash --signHash --encrypt --backupID 123 \
  --backupDirectory /tmp/backup
```

18.3.2.3 Performing an Incremental Backup on All Back Ends

Incremental backups save only those changes that have occurred since the last backup (full or incremental). The main advantage of an incremental backup is the faster time to back up a system when compared to that of full backups. The disadvantage of an incremental backup is that each incremental backup must be restored, which requires more time and care than that of a full restore.

To perform an incremental backup, run the backup command with the `--incremental` option, as follows:

```
$ backup --backUpAll --incremental --compress --backupDirectory /tmp/backup
```

18.3.2.4 Backing Up a Specific Back End

You can back up a single back end by using the `--backendID` option, which specifies the back end to save.

1. List the back ends that are configured on the server, by running the `list-backends` command. For example:

   ```
   $ list-backends

   Backend ID     Base DN
   -------------- -----------------
   adminRoot      cn=admin data
   ads-truststore cn=trust-store
   backup         cn=backups
   config         cn=config
   monitor        cn=monitor
   schema         cn=schema
   tasks          cn=tasks
   userRoot       dc=example,dc=com
   ```

2. Run the backup command with the `--backendID` option.

   For example, to back up the `userRoot` back end, run the following command:

   ```
   $ backup --backendID userRoot --backupDirectory /tmp/backup
   ```

   If you back up a single back end and replication is configured, any changes that you make to that back end are stored in the change log on the replication server. When you restore that back end, the replication server detects that the back end is not up to date and replays the changes made after the backup. This behavior occurs even if there is only one directory server in the replicated topology, because the changes are stored on the replication server.

   If you do not want this behavior, back up all back ends in a replicated environment. This ensures that the data, and the replication server are backed up. In this case when a restore is done, the directory server and the replication server are restored to their state before the back up, and no memory of subsequent changes remains.

18.3.2.5 Performing an Incremental Backup on a Specific Back End

Perform an incremental backup on a specific backend following these steps:

1. List the back ends that are configured on the server, by running the `list-backends` command. For example:

```
$ list-backends

Backend ID     Base DN
-------------- -----------------
adminRoot      cn=admin data
ads-truststore cn=trust-store
backup         cn=backups
config         cn=config
monitor        cn=monitor
schema         cn=schema
tasks          cn=tasks
userRoot       dc=example,dc=com
```

2. Run the backup command with the `--incremental` option.

```
$ backup --incremental --backendID userRoot --backupDirectory /tmp/backup
```

18.3.2.6 Scheduling a Backup as a Task

The directory server provides a task back end for processing administrative tasks, such as backups and restores. You can specify the start time for a backup or restore by using the `-t` or `--start` option. If one of these options is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time. If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Access to the task back end is provided over SSL through the administration connector. If you schedule the backup as a task, you must therefore specify how the SSL certificate will be trusted. This example schedules a backup for execution at a future time. The `-X` option specifies that all certificates presented by the server are trusted. For more information, see Managing Administration Traffic to the Server.

1. Run the `backup` command with the following options:

```
$ backup --port 4444 --bindDN "cn=Directory Manager" \
  --bindPasswordFile pwd-file -X \
  --backUpAll --backupDirectory /tmp/backups --start 20080601121500 \
  --completionNotify admin@example.com --errorNotify admin@example.com
```

2. View information about the scheduled task by using the `manage-tasks` command. For example:

```
$ manage-tasks --port 4444 --bindDN "cn=Directory Manager" \
  --bindPasswordFile pwd-file -X --info 2008040210324704 --no-prompt
```

18.3.3 About the Server Configuration Back Up

All configuration settings for a directory server instance are stored in the `config.ldif` file, which is located in the `config` directory. The directory server automatically saves the `config.ldif` file to ensure that changes are properly accounted for in the configuration.

The file is saved at two specific times:

- **At startup**. If the current configuration does not match the archived configuration, the server saves the `config.ldif` file.
- **At modification time**. Whenever a directory administrator makes changes to the configuration by using the `dsconfig` utility with the server online, the directory server saves the `config.ldif` file prior to the change.

You can access archived configuration files from the `INSTANCE_DIR/OUD/config/archived-configs` directory. This directory lists each saved configuration file, compresses it as a `.gz` file, and saves the configuration as `config-timestamp.gz`. For example, you can see archived `config.ldif` files as follows:

```
$ ls config/archived-configs
09/02/2010 03:43 PM 9,045 config-20100819055359Z.gz
```

18.3.4 Backing Up the Directory Server for Disaster Recovery

Directory and system administrators should have a disaster recovery plan in place in case a natural, human-induced, or catastrophic disaster occurs. If your directory service is distributed over multiple individual servers, back up all the servers individually or back up all the directory data from a central location.

Alternatively, consider replication as a backup and restore strategy. Replication provides faster restores and more update data from another replicated server. For more information, see Considerations for Re-instating Replicated Directory Servers.

The following procedure explains how to back up the Directory Server for Disaster Recovery:

1. Make a backup of all back ends by using the `--backUpAll` option, for example:

   ```
   $ backup --backUpAll --backupDirectory /tmp/backup
   ```

2. Copy the configuration directory, `INSTANCE_DIR /OUD/config`.

   Ensure that the `schema` subdirectory is present within the `INSTANCE_DIR /OUD/config` directory.

3. Copy the files in `INSTANCE_DIR/OUD/logs`.

4. Make a copy of the installation directory.

5. Store the archived data, configuration directory, schema subdirectory, log files and installation directory together in a single location.

   All items are required when restoring the server.

18.3.5 Backing up and Restoring Data Using File System Snapshots

For certain deployments, file system snapshot technologies offer a viable alternative to the traditional backup. On Solaris systems, ZFS enables file system snapshots that are space efficient, very quick to create, and portable between systems. By dedicating a Directory Server per data center, or two if your entire service runs in one data center, you deploy an effective, redundant solution for restoring data as part of your disaster recovery plan.

This section contains the following topics:

18.3.5.1 Taking a ZFS Snapshot On a Dedicated Backup Server

The following procedure explains how to take a ZFS snapshot on a dedicated server:

1. Because the Directory Server is dedicated to backup, configure the server as a read-only replica if you have not already done so.

   ```
   $ dsconfig -h host -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
     set-global-configuration-prop --set writability-mode:internal-only
   ```

When you restore a server from the snapshot of the read-only replica, the restored server accepts only replication traffic until you enable writability, after the server has caught up with other replicas in the topology.

2. Take the ZFS snapshot.

   For example, if the Directory Server files are stored in the file system corresponding to `zpool/DS_FS`, the command is:

   ```
   $ zfs snapshot zpool/DS_FS@{todays_date}
   ```

3. Back up the snapshot to other storage.

   ```
   $ zfs send zpool/DS_FS@{today_date} > /backups/DS_FS.{today_date}.zfs
   ```

   Do not keep snapshots longer than the replication purge delay, because when you restore from a snapshot, the replication mechanism has to be able to replay all the missing changes on the replica.

18.3.5.2 Re-instating a Directory Server From a ZFS Snapshot

The following procedure explains how to re-instate a Directory Server from a ZFS snapshot:

1. Import the backup `zpool`.

   Create a ZFS file system to access the backup pool, using `/backups` as the mount point.

2. Stop the Directory Server that is being re-instated.
3. Initialize the ZFS file system from `/backups`.

   ```
   $ dd if=/backups/DS_FS.{date_to_re-instate}.zfs bs=32k | zfs receive -F zpool/DS_FS
   ```

4. Adapt the configuration as necessary to use the host name and port numbers of the Directory Server to re-instate.
5. Start the Directory Server.
6. Monitor replication until you observe that the Directory Server is synchronized with other replicas in the topology.
7. Set the `writability-mode` to `enabled`, allowing the Directory Server to process write operations from clients.

   ```
   $ dsconfig -h restored-host -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
     -n set-global-configuration-prop --set writability-mode:enabled
   ```

18.3.6 Restoring Data

You can restore data by using the `restore` utility. The `restore` utility allows you to restore only one back end at a time. The directory server must be stopped prior to a restore, unless you are scheduling a restore task, or you are restoring data that has been signed or hashed.

This section contains the following topics:

- Restoring a Back End
- Restoring a Back End From Incremental Backups
- Scheduling a Restore as a Task
- Restoring the Configuration File
- Restoring a Directory Server During Disaster Recovery

18.3.6.1 Restoring a Back End

The following procedure explains how to restore a back end:

1. Stop the server, if it is running.
2. Display the backup information by running the `restore` command with the `--listBackups` option. For example:

```
$ restore --listBackups --backupDirectory backup/userRoot
Backup ID: 20080827153501Z
Backup Date: 27/Aug/2008:10:35:11 -0500
Is Incremental: false
Is Compressed: true
Is Encrypted: false
Has Unsigned Hash: false
Has Signed Hash: false
Dependent Upon: none
```

3. Restore the back end.

```
$ restore --backupDirectory backup/userRoot
```

4. Repeat the restore for the other back ends.

18.3.6.2 Restoring a Back End From Incremental Backups

Typically, system administrators run a weekly full backup with daily incremental backups. Be aware that it takes longer to restore your system from incremental backups.

1. Restore the last full backup on your system by using the `restore` command.

   Each back end must be restored individually.
2. Restore each incremental backup by using the `restore` command.

   Restore each incremental backup starting from the last full backup.

18.3.6.3 Scheduling a Restore as a Task

The directory server provides a task back end for processing administrative tasks, such as backups and restores. You can specify the start time for a restore by using the `-t` or `--start` option. If one of these options is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time. If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Access to the task back end is provided over SSL, using the administration connector. If you schedule the restore as a task, you must therefore specify how the SSL certificate will be trusted.

1. Ensure that the server is stopped prior to the scheduled restore time.
2. Schedule the restore by using the `-t` or `--start` option of the `restore` command.

   The following command restores the `userRoot` back end at a scheduled start time by using the `--start` option. The restore sends a completion and error notification to `admin@example.com`. The `-X` option specifies that all certificates presented by the server are trusted.

```
$ restore -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  -d /backup/userRoot --start 20080125121500 --completionNotify admin@example.com \
  --errorNotify admin@example.com
```

3. You can view this scheduled task by using the `manage-tasks` utility.

   For more information, see Configuring Commands As Tasks.

18.3.6.4 Restoring the Configuration File

You might need to restore the configuration file to transfer the configuration to another server, for disaster recovery purposes, or for other events. In general, if a server is online, the current configuration file is equivalent to the latest archived configuration file. However, you can choose to restore the `config.ldif` file from a previous date.

1. Stop the server if it is running.
2. Locate the required configuration file on the system. For example:

   ```
   $ ls INSTANCE_DIR/OUD/config/archived-configs
   ./
   ../
   config-20110817192057Z.gz
   config-20110827153200Z.gz
   config-20110817192052Z.gz
   config-20110827153214Z-2.gz
   ```

3. Manually decompress the archived configuration file, using a decompression utility such as `gunzip`.
4. Copy the file to the `config` directory, replacing the current `config.ldif` file.

   ```
   $ cp config-20110817192057Z INSTANCE_DIR/OUD/config/config.ldif
   ```

18.3.6.5 Restoring a Directory Server During Disaster Recovery

The following procedure explains how to restore a Directory Server during disaster recovery:

1. Install the same version of the directory server that was previously installed on the host.
2. Create a server instance by using the `setup` command.
3. Copy the saved `config` directory to `INSTANCE_DIR/OUD/config`.

   The `config.ldif` file should reside in this directory. The saved schema subdirectory should be located in `INSTANCE_DIR/OUD/config/schema`.
4. Check that the configuration for the restored server is correct.
5. Restore the individual back ends by using the `restore` command.

18.3.7 Considerations for Re-instating Replicated Directory Servers

Performing binary restores in replicated environments requires special care depending on your replicated topology. If possible, update your back end by using the replication mechanisms in your system instead of re-instating it from a backup. Replication has distinct advantages over traditional tape backups. Data restores are much faster than tape restores, and the data is more up to date. However, tapes are still needed in the event that the replicated data is corrupt and has been propagated to other servers.

When re-instating a replicated server, ensure that the configuration file `INSTANCE_DIR/OUD/config/config.ldif` is the same as when the backup was made. Re-instate the `config.ldif` file prior to re-instating the server back ends.

You cannot re-instate an old backup to a master server because it might be out of date. Rather allow the replication mechanism to bring a master up to date with the other master servers by setting that master to read-only. When the master has been synchronized, you can reset it to read/write.

If you must re-instate a replicated server, re-initialize the server from one of the other replicated servers by importing an LDIF file.

For very large databases (millions of entries), make a binary copy of one server and re-instate it on the other replicated server.

If you have a fairly recent backup (one that is not older than the maximum age of the change log contents on any of the other replicated servers), you can use that version to re-instate your data using the `restore` command. When the old backup is re-instated, the other servers will update that server with recent updates made since the backup was saved. See Changing the Data Set in an Existing Replicated Topology for more information on replication changelogDB.

18.3.8 Deleting Backup Data Files

By running regular backups, the backup files might start to consume too much disk space. You must remove the old backup files manually to create space for new ones.

> **Note:**
>
> You can purge backup data files automatically using the `purge-backup` utility. This can be done both offline and online. Based on the purge criteria set, it automatically trims the backup.info file and deletes the backup data files. See Purging Backup Data Files Automatically.

When you delete backup files manually, ensure that you do not break any dependencies between backup sets.

Perform the following procedure to delete the backup data:

1. List the existing backups in your backup directory.

   For example, to list the backups in the default backup directory, run the following command:

   ```
   UNIX: $ ls INSTANCE_DIR/OUD/bak
          backup-userRoot-20110929184101Z backup-userRoot-20111029184509Z
          backup.info backup.info.save

   WINDOWS: C:\> dir INSTANCE_DIR\OUD\bak
            backup-userRoot-20110929184101Z backup-userRoot-20111029184509Z
            backup.info backup.info.save
   ```

2. Delete the backup file from the backup directory.

   For example, to remove the oldest backup of the `userRoot` database in the preceding step, run the following command:

   ```
   UNIX: $ rm INSTANCE_DIR/OUD/bak/backup-userRoot-20110929184101Z

   WINDOWS C:\> del INSTANCE_DIR\OUD\bak\backup-userRoot-20110929184101Z
   ```

3. Remove the associated backup information from the `backup.info` file.

   You can display the contents of the `backup.info`, as follows (on UNIX systems):

```
$ more INSTANCE_DIR/OUD/bak/backup.info
backend_dn=ds-cfg-backend-id=userRoot,cn=Backends,cn=config

     backup_id=20110929184101Z
     backup_date=20110929184104Z
     incremental=false
     compressed=false
     encrypted=false
     property.last_logfile_name=00000000.jdb
     property.last_logfile_size=160773
     property.archive_file=backup-userRoot-20110929184101Z

     backup_id=20111029184509Z
     backup_date=20111029184512Z
     incremental=false
     compressed=false
     encrypted=false
     property.last_logfile_name=00000000.jdb
     property.last_logfile_size=160773
     property.archive_file=backup-userRoot-20111029184509Z
```

   For Windows systems, use an appropriate text editor.

18.3.9 Purging Backup Data Files Automatically

The directory server provides an effective command-line utility, `purge-backup`, for automatically purging backup data files.

The `purge-backup` command can be executed immediately or as a scheduled job. If the `purge-backup` is scheduled, the command contacts the server over SSL, using the administration connector, and registers a `purge-backup` task. If no connection options are specified, the command runs immediately. Based on the purge criteria set, this job automatically trims the backup.info file and deletes the backup data files.

The following procedures demonstrate how to utilize the `purge-backup` command in a variety of scenarios:

- Purging Backup Data for All Back Ends
- Purging Backup Data for Specific Back Ends
- Scheduling a Purge-Backup as a Task

18.3.9.1 Purging Backup Data for All Back Ends

You can purge the backup data files for all back ends using the `--purgeAll` option. You can use this option when a prior backup has been executed using the `--backUpAll` option.

See Backing Up All Back Ends for a list of back ends for which backup is supported. The same is applicable for purging as well.

All `purge-backup` operations expect a `--purgeDelay` argument, which defines the number of days before which the backup data is purged. The backup directory is anticipated to have subdirectories for each back end, with each subdirectory including the backup.info file as well as the backup data files.

Consider the following example. Here you run the following command on a standalone directory server, which specifies that backup data for all back ends older than 30 days should be considered for purging in the specified backup location.

```
./purge-backup --backupDirectory /tmp/backup --purgeDelay 30 --purgeAll
```

The utility trims the backup.info file by eliminating all backup info sets that have qualified for purge depending on the `purgeDelay` parameter, and then deletes the backup files from the backup directory. You must bear in mind that the most recent backup set will not be considered for purge.

Following the completion of the `purge-backup` operation, the contents of /tmp/backup/userRoot would be as follows:

```
$ls /tmp/backup/userRoot/
backup.info  backup-userRoot-20230924134151Z

backup.info would look like:

backend_id=userRoot

backup_id=20230924134151Z
backup_date=20230924134155Z
incremental=false
compressed=true
encrypted=false
property.archive_file=backup-userRoot-20230924134151Z
property.last_logfile_size=37964
property.last_logfile_name=00000000.jdb
```

18.3.9.2 Purging Backup Data for Specific Back Ends

You can purge backup data for a single back end using the `--backendID` option.

Run the `listbackends` command to see a list of the back ends that are configured on the server. See Backing Up a Specific Back End.

To purge backup data for the userRoot back end, execute the following command:

```
purge-backup --backendID userRoot --backupDirectory /tmp/backup/userRoot --
purgeDelay 10
```

18.3.9.3 Scheduling a Purge-Backup as a Task

You can periodically purge old backup files. The directory server provides a task back end for processing administrative tasks, such as `purge-backup`.

You can specify the start time for a `purge-backup` by using the `-t` or `--start` option. If one of these options is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time. If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Access to the task back end is provided over SSL through the administration connector. If you schedule the `purge-backup` as a task, you must therefore specify how the SSL certificate will be trusted. This example schedules a `purge-backup` for execution at a future time. The `-X` option specifies that all certificates presented by the server are trusted. See Managing Administration Traffic to the Server.

Run the `purge-backup` command with the following options:

```
purge-backup --port 4444 --bindDN "cn=Directory Manager" \
--bindPasswordFile pwd-file -X \
--purgeAll \
--purgeDelay 30 \
--backupDirectory /tmp/backups
--start  20230924131502417
```

View information about the scheduled task by using the `manage-tasks` command. For example:

```
manage-tasks --port 4444 --bindDN "cn=Directory Manager" \
--bindPasswordFile pwd-file -X --info 20230924131220034 --no-prompt
```

18.4 About Searching Directory Data

The directory server provides a suite of LDAPv3-compliant command-line tools, including a sophisticated look-up operation in the form of a search function and filters. You can also use Oracle Directory Services Manager to search directory data.

This section explains how to use the `ldapsearch` command-line utility and Oracle Unified Directory Services Manager to locate entries in the directory.

This section contains the following topics:

- Overview of the `ldapsearch` Command
- About ldapsearch Location and Format
- Understanding Search Criteria
- Using ldapsearch Command
- Searching Data Using OUDSM

18.4.1 Overview of the `ldapsearch` Command

The `ldapsearch` command allows you to enter a search request where you specify the host name, port, bind DN and password plus search criteria to locate entries in the directory. When an LDAP client makes a search request to the directory server, it opens a connection to the directory server over TCP/IP. The client then performs a *bind* operation to the directory server by attempting to match a given entry, which effectively authenticates the client. Most users have the option to bind as a particular user, such as a Directory Administrator or themselves, or to not bind as any user, in which case the directory server assumes that the user is bound as an *anonymous* user.

Because all access to directory data is based on how a connection is bound, the directory server checks the client's privileges to see if the client can run a particular search operation. After the directory server checks the user's access rights, the client passes a search request consisting of a set of search criteria and options to the directory server.

The directory server searches all entries that match the search criteria and options. It then returns the entries, the DN, and all attributes for each entry, in the form of LDIF text to standard output. If an error occurs, the directory server displays an error message indicating the error. Finally, the client closes the connection when the search operation has completed.

18.4.2 About ldapsearch Location and Format

Learn about the ldapsearch location and the format. Understand about the purpose of each option in the format.

The `ldapsearch` utility is found in the following location:

```
(UNIX, Linux) INSTANCE_DIR/OUD/bin
(Windows) INSTANCE_DIR\OUD\bat
```

The utility has the following format:

```
ldapsearch optional-options search-filter optional-list-of-attributes
```

where:

- *optional-options* are command-line options that must appear before the search filter.
- *search-filter* is an LDAP search filter either specified on the command-line or in a file.
- *optional-list-of-attributes* is a list of attributes separated by a space. The list of attributes must appear after the search filter.

About Common `ldapsearch` Options explains about the common ldapsearch options in detail.

18.4.2.1 About Common `ldapsearch` Options

The `ldapsearch` command has many options to search entries in the directory. Options are allowed in either their short form (for example, `-b baseDN`) or their long form (for example, `--baseDN`). The most common command options to use with `ldapsearch` are as follows:

`-h, --hostname address`
Specifies the host name or IP address of the directory server on which the search should be run. It can be an IP address or a resolvable name. If this is not provided, a default value of `localhost` is used.

`-p, --port port`
Specifies the directory server port. It should be an integer value between `1` and `65535`, inclusive. If this is not provided, a default port of `389` is used.

`-b, --baseDN baseDN`
Specifies the base DN to use for the search operation. If a file containing multiple filters is provided using the `--filename` option, this base DN is used for all of the searches. This is a required option.

`-s, --searchScope scope`
Sets the scope for the search operation. Its value must be one of the following:

- `base`. Searches only the entry specified by the `--baseDN` or `-b` option.
- `one`. Searches only the entry specified by the `--baseDN` or `-b` option and its immediate children.

- `sub` or `subordinate`. Searches the entire subtree whose base is the entry specified by the `--baseDN` or `-b` option. This is the default option when no `--searchScope` option is provided.

`-D, --bindDN bindDN`
Specifies the DN to use when binding to the directory server through simple authentication. This option is not required when using SASL authentication or anonymous binding.

`-w, --bindPassword bindPassword`
Specifies the password to use when binding to the directory server. This option is used for simple authentication, as well as for password-based SASL mechanisms like CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if anonymous binding is used. This option must not be used with the `--bindPasswordFile` option. To prompt for the password, type `-w -`.

`-l, --timeLimit numSeconds`
Sets the maximum length of time in seconds that the directory server should spend processing any search request. If this is not provided, no time limit is imposed by the client.

> **Note:**
>
> The directory server may enforce a lower time limit than the one requested by the client.

`-z, --sizeLimit numEntries`
Sets the maximum number of matching entries that the directory server should return to the client. If this is not provided, no maximum size is imposed by the client.

> **Note:**
>
> The directory server may enforce a lower size limit than the one requested by the client.

`-S, --sortOrder sortOrder`
Sorts the results before returning them to the client. The sort order is a comma-delimited list of sort keys, where each sort key consists of the following elements:

- `+/- (plus or minus sign)`. Indicates that the sort should be in ascending (+) or descending (-) order. If this value is omitted, the sort uses ascending order by default.
- `Attribute name`. The name of the attribute to sort the data. This element is required.
- `Name` or `OID Matching Rule`. An optional colon followed by the name or OID of the matching rule used to perform the sort. If this is not provided, the default ordering matching rule for the specified attribute type is used.

  For example, the sort order string `sn,givenName` sorts the entries in ascending order first by `sn` and then by `givenName`. Alternately, using `-modifyTimestamp`, the directory server sorts the `modifyTimestamp` attributes with the most recent values first.

18.4.3 Understanding Search Criteria

Learn about the search criteria and the various filter options.

This section contains the following topics:

18.4.3.1 Overview of Search Criteria

The `ldapsearch` command requires three sets of information to specify where and what to search in the directory information tree:

- **Base DN**. By specifying the base DN, you are defining the topmost distinguished name (DN) or starting point in the directory to conduct the search. All searches begin at or below the base DN, depending on the scope, and move down the tree, never upwards. Examples of base DNs are: `dc=example,dc=com` and `ou=People,dc=example,dc=com`.
- **Scope**. The scope determines which set of entries at or below the base DN should be evaluated by the search filter. The search scope and base DN together indicate "where" to look for entries in the directory.
- **Search filter**. The search filter specifies the conditions that the entries must meet to be returned to the client.

18.4.3.2 Search Filter Types and Operators Specifications

The directory server provides seven types of search filters, defined in the LDAP protocol. With each search filter type, you use operators that test the relationships between two entities, *attribute* and *value*.

The following table shows how search filters are used to return specific entries in a search query.

Search Filter	Operator	Description
Presence	`attr=*`	Return all entries that have any value associated with the specified attribute. The filter uses the wildcard character to denote zero or more characters in the string. For example, the following filter is common and returns all entries that have an object class with any value, which every entry has: `(objectclass=\*)`. **Note:** the LDAP protocol specifies that filters should have the form "(filter)", which includes parentheses surrounded by quotation marks. Although most directory servers accept filters without the parentheses and quotation marks, it is good practice to include them.
Equality	`attr=value`	Return entries containing attributes equal to a specified value. For example: `(sn=Bergin)` returns all entries that have a surname (`sn`) attribute with the value of `Bergin`. **Note:** The `sn` value is case insensitive, so entries associated with `sn=bergin` or `sn=Bergin` will be returned.

Search Filter	Operator	Description
Substring	`attr=<initial-string><any substring> <final-string>`	Return entries with attributes containing a specified substring or partial substring. The filter uses the wildcard character to denote zero or more characters in the string. • Run an initial substring search that looks for all attribute values that have the characters `Ber` at the start of the string: `(sn=Ber\*)` • Specify the middle substring of an attribute value. For example: `(sn=\*erg\*)` • Specify the end of a substring of an attribute value. For example: `(sn=\*gin)`. Or you can specify some combination of substrings • Specify the initial and middle substring: `(sn=ber\*gi\*)` • Specify the initial and ending substrings: `(sn=be\*in)` • Specify the middle and end substrings: `(sn=\*er\*in)` **Note:** Substring filters do not use true wild cards such as in system listings or regular expressions. Thus, the following filter would be invalid because of too many criteria: `(sn=\*B\*rg\*n)`.
Greater than or equal to	`attr>=value`	Return entries containing attributes that are greater than or equal to the specified value. For example, `(sn>=Bergin)` returns all entries that have an attribute greater than or equal to the value, `Bergin`, based on the matching rules for attributes (see Overview of Matching Rules).
Less than or equal to	`attr<=value`	Return entries containing attributes that are less than or equal to the specified value. For example, `(sn<=Bergin)` returns all entries that have an attribute less than or equal to the value, `Bergin`, based on the matching rules for attributes.
Approximate	`attr~=value`	Return entries containing the specified attribute with a value that is approximately equal to the value specified in the search filter. For example: `(sn~=Bergan)` could return the entry associated with `(sn=Bergin)` or `(sn=Bergan)`. The Approximate search filter works only with English language strings. It does not work with non-ASCII-based strings, such as Ja or Zn.
Extensible match	`attr= attr [":dn"] [":" matchingrule] ":=" value Or:[:dn] ":" matchingrule ":=" value`	Return the results entries when an attribute equals the value with the specified matching rule. LDAP version 3 enables you to build match operators and rules for a particular attribute. Matching rules define how to compare attribute values with a particular syntax. In other words, an extensible search filter enables you to add a matching rule to a search filter. For example, the following search filter compares entries containing the surname attribute with value equal to "`Jensen`" by using the matching rule designated by OID 2.5.13.5: `(sn:2.5.13.5:=Jensen)`. Another example illustrates the use of the "`:dn`" notation to indicate that the OID `2.5.13.5` should be used when making comparisons, and that the attributes of an entry should be considered part of the entry when evaluating the match: `(sn:dn:2.5.13.5:=Jensen)`.

18.4.3.3 Compound Search Filters Evaluation

Multiple search filter components can be combined and evaluated by using the operator:

`(`*`Boolean-Operator`*`(`*`filter`*`)(`*`filter`*`)(`*`filter`*`))`

Boolean operators can be combined and nested together to form complex expressions:

`(`*`Boolean-Operator`*`(`*`filter`*`)(`*`Boolean-operator`*`(`*`filter`*`)(`*`filter`*`)))`

The following table describes the Boolean operators.

Search Filter	Operator	Description
AND	`(&(filter)(filter))`	All specified filters must be true for the statement to be true. For example, `(&(sn=Carter)(l=Cupertino))` returns all entries that have the surname attribute equal to "Carter" and the location attribute equal to `Cupertino` if any.
OR	`(|(filter)(filter))`	At least one specified filter must be true for the statement to be true. For example, `(|(sn=Carter)(l=Cupertino))` returns all entries that have the surname attribute equal to `Carter` or the location attribute equal to `Cupertino` if any.
NOT	`(!(filter)(filter))`	The specified filter must not be true for the statement to be true. For example, `(!(sn=Bergin))` returns all entries that do *not* have a surname attribute equal to the string `Smith`. The filter also returns all entries that do not have the `sn` attribute.

18.4.3.4 Using UTF-8 Encoding in Search Filters

UTF8 is a byte-order, variable-length character code for Unicode and a subset of ASCII. You use UTF-8 for multiple-language support by replacing each character of a non 7-bit ASCII character with a byte of a UTF-8 encoding. Typically, you must escape the UTF-8 encoding with a backslash.

For example, the character `é` has a UTF-8 representation of `c3a9` and `è` has a UTF-8 representation `c3a8`. A UTF-8 encoding is represented with an escaped backslash. So, `é` is represented as `\\c3\\a9` and `è` is represented as `\\c3\\a8`. To represent `cn=Hélène Laurent`, you would use the following encoding:

```
(cn=H\\c3\\a9l\\c3\\a8ne Laurent)
```

18.4.3.5 Special Characters in Search Filters

You must specify special characters (for example, a space, backslash, asterisk, comma, period, or others) by using the escape backslash.

- Asterisk. Represent an asterisk (`*`) as `\\2a`. For example, `Five*Star` would be represented as "`(cn=Five\\2aStar)`".
- Backslash. Represent a backslash (`\`) as `\\5c`. For example, `c:\\file` would be represented as "`(cn=c:\\5c\\5cfile)`".
- Parentheses. Represent parentheses `()` as `\\28` and `\\29`, respectively. For example, `John Doe (II)` would be represented as "`(cn=John Doe \\28II\\29)`".
- Null. Represent null as `\\00`. For example, 0001 would be represented as "`(bin=\\00\\00\\00\\01)`".
- Comma. Represent a comma (`,`) by escaping it as `\\,`. For example, "`(cn=Mkt\\,Peru,dc=example,dc=com)`".
- Space. Generally, use quotation marks around strings that contain a space. For example, `(cn="HR Managers,ou=Groups,dc=example,dc=com")`.

18.4.4 Using ldapsearch Command

You can use `ldapsearch` command to search for specific user attributes, perform search with different scopes, return attributes and entries. Learn about using `ldapsearch` command from the following topics.

This section contains the following topics that explain the use of`ldapsearch` command:

- About ldapsearch Command Options
- Returning All Entries
- Searching For a Specific User
- Searching for Specific User Attributes
- Performing a Search With Base Scope
- Performing a Search With One-Level Scope
- Performing a Search With Subtree Scope
- Returning Attribute Names Only
- Returning User Attributes Only
- Returning Base DNs Only
- Searching For Specific Object Classes
- Returning A Count of Matching Entries in the Directory
- Performing a Search With a Compound Filter
- Performing a Search Using a Filter File
- Limiting the Number of Entries Returned in a Search

18.4.4.1 About ldapsearch Command Options

The following examples show the use of the `ldapsearch` command with various search options. These examples all assume that your current working directory is `INSTANCE_DIR/OUD/bin` (`INSTANCE_DIR\OUD\bat` on Windows systems).

The following points pertain to all the examples in this section:

- If the example does not specify a scope (with the `--searchScope` or `-s` option), `ldapsearch` assumes that the scope is `subordinate` or `sub`, which returns the full subtree of the base DN.
- If no attributes are specified, the command returns all attributes and their values.
- If no `--bindDN` and `--bindPassword` are specified, the search uses an anonymous bind.
- If no `--hostname` is specified, the default (`localhost`) is used.

> **Note:**
>
> Many UNIX and Linux operating systems provide an installed version of common LDAP-client tools, such as `ldapsearch`, `ldapmodify`, and `ldapdelete` in the `/usr/bin` directory. You should use the `ldapsearch` provided with the directory server to search the directory server. You can check which version of `ldapsearch` you are using by typing the following command:
>
> ```
> $ which ldapsearch
> ```
>
> If you are using the `ldapsearch` in `/usr/bin`, put `INSTANCE_DIR/OUD/bin` at the beginning of your `$PATH`.

18.4.4.2 Returning All Entries

You can return all entries below a specified branch DN using the presence search filter `(objectclass=*)`. The search filter looks for all entries that have one or more object classes with any value. Because all entries have several object class definitions, the filter guarantees that all entries will be returned.

Run the `ldapsearch` command with the filter `(objectclass=*)`.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example

dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups

dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniquenames
objectClass: top
ou: Groups
cn: Directory Administrators
uniquemember: uid=kvaughan, ou=People, dc=example,dc=com
uniquemember: uid=rdaugherty, ou=People, dc=example,dc=com
uniquemember: uid=hmiller, ou=People, dc=example,dc=com
...
```

18.4.4.3 Searching For a Specific User

You can use an equality filter to locate a specific user in the directory. This example locates an employee with the common name of "Frank Albers".

Run the `ldapsearch` command with the filter `"(cn=Frank Albers)"`.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com  "(cn=Frank Albers)"

dn: uid=falbers,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```

```
objectClass: top
givenName: Frank
uid: falbers
cn: Frank Albers
sn: Albers
telephoneNumber: +1 408 555 3094
userPassword: {SSHA}nDTQJ9DDiMUrBwR0WNKq0tgS4iB2A9QJFgpZiA==
roomNumber: 1439
ou: Accounting
ou: People
l: Sunnyvale
mail: falbers@example.com
facsimileTelephoneNumber: +1 408 555 9751
```

18.4.4.4 Searching for Specific User Attributes

You can use an equality filter to locate an entry's attribute(s) in the directory. Specify one or more attributes by placing them after the search filter. This example locates the `telephoneNumber` and `mail` attributes from the user entry for `Frank Albers`.

Run the `ldapsearch` command with the filter `"(cn=Frank Albers)"` and the corresponding attributes.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com \
 "(cn=Frank Albers)" telephoneNumber mail
dn: uid=falbers,ou=People,dc=example,dc=com
telephoneNumber: +1 408 555 3094
mail: falbers@example.com
```

18.4.4.5 Performing a Search With Base Scope

Together with the search base DN, the scope determines what part of the directory information tree (DIT) is examined. A base scope examines only the level specified by the base DN (and none of its child entries). You specify a base scope by using the `--searchScope base` option or its short form equivalent `-s base`.

Run the `ldapsearch` command with the `--searchScope base` option.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope base "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

18.4.4.6 Performing a Search With One-Level Scope

A one-level scope examines only the level immediately below the base DN. You specify a one-level scope by using the `--searchScope one` option or its short form equivalent `-s one`. This example displays the entries immediately below the base DN.

Run the `ldapsearch` command with the `--searchScope one` option.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope one "(objectclass=*)"
dn: ou=Groups,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
ou: Groups
dn: ou=People,dc=example,dc=com
```

```
objectClass: top
objectClass: organizationalunit
ou: People
dn: ou=Special Users,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
ou: Special Users
description: Special Administrative Accounts
dn: ou=Company Servers,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
ou: Company Servers
description: Standard branch for Company Server registration
```

18.4.4.7 Performing a Search With Subtree Scope

The subtree scope examines the subtree below the base DN and includes the base DN level. You specify a subtree scope using the `--searchScope sub` option, or its short form equivalent `-s sub`. If you do not specify the `--searchScope`, `ldapsearch` assumes a subtree scope.

Run the `ldapsearch` command with the `--searchScope sub` option.

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "cn=Directory Administrators,ou=Groups,dc=example,dc=com" \
  --searchScope sub "(objectclass=*)"
dn: cn=HR Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage HR entries
cn: HR Managers
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=cschmith, ou=People, dc=example,dc=com
```

18.4.4.8 Returning Attribute Names Only

The `ldapsearch` command provides a convenient option to check if an attribute is present in the directory. Use the `--typesOnly` option or its short form equivalent `-A` to instruct the directory server to display the attribute names but not their values.

Run the `ldapsearch` command with the `--typesOnly` option.

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "dc=example,dc=com" --typesOnly "(objectclass=*)"
dn: dc=example,dc=com
objectClass
dc
dn: ou=Groups,dc=example,dc=com
objectClass
ou ...
```

18.4.4.9 Returning User Attributes Only

You can use `ldapsearch` to return only user attributes for entries that match the search filter, by including an asterisk `*`. User attributes (as opposed to operational attributes) store user information in the directory. If you do not specify the asterisk, the user attributes are returned by default. You must escape the asterisk appropriately for your shell.

Run the `ldapsearch` command, specifying `'*'` after the search filter.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)" '*'
dn: cn=Aggie Aguirre,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: top
postalAddress: Aggie Aguirre$15172 Jackson Street$Salt Lake City, MI  49843
postalCode: 49843
uid: user.99
description: This is the description for Aggie Aguirre.
employeeNumber: 99
initials: AGA
givenName: Aggie
pager: +1 514 297 1830
mobile: +1 030 300 0720
cn: Aggie Aguirre
telephoneNumber: +1 730 027 2062
sn: Aguirre
street: 15172 Jackson Street
homePhone: +1 229 128 3072
mail: user.99@maildomain.net
l: Salt Lake City
st: MI
```

18.4.4.10 Returning Base DNs Only

You can use `ldapsearch` to return only the base DNs for entries that match the search filter by including a `1.1` string after the search filter.

Run the `ldapsearch` command, specifying `1.1` after the search filter.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)" 1.1
version: 1
dn: cn=Richard Arnold,ou=people,dc=example,dc=com

dn: cn=Kevin Booysen,ou=people,dc=example,dc=com

dn: cn=Steven Morris,ou=people,dc=example,dc=com

dn: cn=Leila Shakir,ou=people,dc=example,dc=com

dn: cn=Emily Smith,ou=people,dc=example,dc=com
...
```

18.4.4.11 Searching For Specific Object Classes

You can search all entries where the attributes are referenced by a specific object class by prepending a `@` character to the object class name. For example, to view all entries that have an object class of `groupOfUniqueNames`, include `@groupOfUniqueNames` after the search filter.

Run the `ldapsearch` command, specifying `@` and the object class after the search filter.

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "ou=Groups,dc=example,dc=com" "(objectclass=*)" @groupOfUniqueNames
dn: ou=Groups,dc=example,dc=com
ou: Groups
objectClass: organizationalunit
objectClass: top
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
```

```
ou: Groups
objectClass: groupofuniquenames
objectClass: top
cn: Directory Administrators
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=rdaugherty, ou=People, dc=example,dc=com
uniqueMember: uid=hmiller, ou=People, dc=example,dc=com ...
```

18.4.4.12 Returning A Count of Matching Entries in the Directory

The `ldapsearch` command provides the `--countEntries` to return the total number of matching entries returned by the directory server. The directory server returns all entries that match the search filter and displays the total number on the last line. This example determines the number of employee entries whose location is Cincinnati.

Run the `ldapsearch` command with the `--countEntries` option.

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --baseDN dc=example,dc=com --countEntries "l=Cincinnati"
dn: cn=Adi Adamski,ou=People,dc=example,dc=com
...
l: Cincinnati
st: OH

dn: Aggi Aguinsky,ou=People,dc=example,dc=com
objectClass: person
...
l: Cincinnati
st: OH

# Total number of matching entries: 2
```

18.4.4.13 Performing a Search With a Compound Filter

Compound search filters involve multiple tests using the boolean operators AND (`&`), OR (`|`), or NOT (`!`). You can combine and nest boolean operators and filters together to form complex expressions. The following example searches for all entries for employees named `Jensen` who work in `Cupertino`. The command returns two results.

Run the `ldapsearch` command with a compound search filter.

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --baseDN dc=example,dc=com "(&(sn=jensen)(l=Cupertino))"
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Development
ou: People
sn: Jensen
...
l: Cupertino
st: CA

dn: uid=rjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
```

```
ou: People
sn: Jensen
...
l: Cupertino
st: CA
```

18.4.4.14 Performing a Search Using a Filter File

You can place complex or multiple filters in a file by using the `--filename` option. If the file contains multiple filters, the file should be structured with one filter per line. Searches are performed using the same connection to the directory server in the order in which they appear in the filter file. If the `--filename` option is used, any trailing options are treated as separate attributes. Otherwise, the first trailing option must be the search filter.

This example searches all entries for employees named `Jensen` who work in `Cupertino` and who do not work in the Accounting department.

1. Create the filter file.

   For this example, create a file called `myfilter.txt` with the following content:
   `(&(sn=jensen)(l=Cupertino)(!(ou=Accounting)))`

2. Run the `ldapsearch` command, specifying the file name as a filter.

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
--bindPassword password --baseDN dc=example,dc=com --filename myfilter.txt
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Development
ou: People
sn: Jensen
l: Cupertino
cn: Barbara Jensen
cn: Babs Jensen
telephoneNumber: +1 408 555 1862
givenName: Barbara
uid: bjensen
mail: bjensen@example.com
```

18.4.4.15 Limiting the Number of Entries Returned in a Search

You can limit the number of entries that are returned by using the `-z` or `--sizeLimit` option. If the number of entries exceeds the number that is specified, the search returns the specified number of entries, then returns an error stating that the size limit was exceeded. The following example requests a maximum of 5 entries.

Run the `ldapsearch` command with the `--sizeLimit` option.

```
$ ldapsearch --hostname localhost --port 1389 -b "dc=example,dc=com" \
  --sizeLimit 5 "objectclass=*" 1.1
dn: dc=example,dc=com

dn: ou=People,dc=example,dc=com

dn: uid=user.0,ou=People,dc=example,dc=com

dn: uid=user.1,ou=People,dc=example,dc=com
```

```
dn: uid=user.2,ou=People,dc=example,dc=com

SEARCH operation failed
Result Code:  4 (Size Limit Exceeded)
Additional Information:  This search operation has sent the maximum of 5 entries to the
client
```

18.4.5 Searching Data Using OUDSM

The Advanced Search tab of each server instance in OUDSM enables you to perform complex searches on directory data.

The following section describes about searching data using OUDSM:

To perform a complex LDAP search using the OUDSM advanced search facility, complete the following steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Advanced Search** tab.
3. Select the appropriate network group from the **Network Group** list.
4. In the **Base Search DN** field, enter the DN that will be the starting point of the search.

   To select an entry as Base Search DN, click **Select**.

   In the **Entry Picker** window, select **Tree View** to navigate the directory tree and locate the entry, or **Search View** to search for the entry.
5. Select the scope of the search from the **Scope** list. The LDAP search scope indicates the set of entries at or below the search base DN that will be considered potential matches for a search operation. The scope can be one of:
   - **Base.** This specifies that the search operation should only be performed against the entry specified as the search base DN. No entries below it will be considered.
   - **One Level.** This specifies that the search operation should only be performed against entries that are immediate subordinates of the entry specified as the search base DN. The base entry itself is not included, nor are any entries below the immediate subordinates of the search base entry.
   - **Subtree.** This specifies that the search operation should be performed against the entry specified as the search base and all of its subordinates to any depth.
6. In the **Filter** field, enter a valid LDAP search filter.

   Alternatively, click **Filter Builder** and enter the required information for OUDSM to build the LDAP search filter.

   For more information about LDAP search filters, see Search Filter Types and Operators Specifications.
7. From the **Search Results Size** list, select how you want OUDSM to limit the number of entries that are returned by the search.
   - **Set Limit** enables you to specify the precise number of entries that are returned.
   - **Use Virtual List View** enables you to use a virtual list view index in the search. For more information, see Searching Using the Virtual List View Control.
   - **Use Paging** enables you to specify that only a subset of the results should be returned at a time, and allows you to indicate the number of results on each page. For more information, see Searching Using the Simple Paged Results Control.

18.5 Using Advanced Search Features

The directory server supports LDAPv3-compliant search functionality by using the `ldapsearch` command. You can use special attributes, security options, and LDAP controls with the search process, based on your system configuration.

This section contains the following topics:

- Searching for Special Entries and Attributes
- Searching Over SSL
- Searching Using Controls
- Searching in Verbose Mode and With a Properties File
- Searching Internationalized Entries
- Sorting Multi-Valued Attributes in a Search Response

For additional information, see About Searching Directory Data, Using a Properties File With Server Commands, and ldapsearch

18.5.1 Searching for Special Entries and Attributes

Learn how to search for operational attributes and Root DSE entry.

This section contains the following topics:

- Searching for Operational Attributes
- Searching the Root DSE Entry
- Searching for ACI Attributes
- Searching the Schema Entry
- Searching the Configuration Entry
- Searching the Monitoring Entry

18.5.1.1 Searching for Operational Attributes

Operational attributes are used for storing information needed for processing by the directory server itself or for holding any other data maintained by the directory server that was not explicitly provided by clients. Operational attributes are not included in entries returned from search operations unless they are explicitly included in the list of search attributes. You can request the directory server to return operational attributes by adding `+` (the plus sign) in your `ldapsearch` command.

Run the `ldapsearch` command with the `+` character.

You must escape the character using a means appropriate to your shell.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -b "dc=example,dc=com" "(objectclass=*)" "+"
...
dn: cn=PD Managers,ou=groups,dc=example,dc=com
numSubordinates: 0
hasSubordinates: false
subschemaSubentry: cn=schema
entryDN: cn=pd managers,ou=groups,dc=example,dc=com
```

```
entryUUID: 38666d52-7a53-332e-902f-e34dd4aaa7a0
...
```

18.5.1.2 Searching the Root DSE Entry

The Root DSE is a special entry that provides information about the server's name, version, naming contexts, and supported features. Because many of the attributes are operational, you must specify `+` (the plus sign) to display the attributes of the Root DSE entry.

Run the `ldapsearch` command with a baseDN of `""`.

Specify the scope as `base` and include the `+` character to display operational attributes.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -b "" --searchScope base "(objectclass=*)" "+"
dn:
supportedExtension: 1.3.6.1.4.1.4203.1.11.3
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.26027.1.6.2
supportedExtension: 1.3.6.1.4.1.26027.1.6.1
supportedExtension: 1.3.6.1.1.8
supportedExtension: 1.3.6.1.4.1.1466.20037
...
```

18.5.1.3 Searching for ACI Attributes

The directory server stores access control instructions (ACIs) as one or more values of the `aci` attribute on an entry to allow or deny access to the directory database. The `aci` attribute is a multi-valued operational attribute that can be read and modified by directory users and that should itself be protected by ACIs. Administrative users are usually given full access to the `aci` attribute and can view its values by running an `ldapsearch` command.

Run the `ldapsearch` command as follows:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -b dc=example,dc=com --searchScope base "(aci=*)" aci
dn: dc=example,dc=com
aci: (target ="ldap:///dc=example,dc=com")(targetattr h3.="userPassword")
  (version 3.0;acl "Anonymous read-search access";allow (read, search, compare)
  (userdn = "ldap:///anyone");)
aci: (target="ldap:///dc=example,dc=com") (targetattr = "*")
  (version 3.0; acl "allow all Admin group"; allow(all)
  groupdn = "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

18.5.1.4 Searching the Schema Entry

The directory server holds schema information in the schema entry (`cn=schema`) for the object classes and attributes defined on your instance.

Run the `ldapsearch` command on the `cn=schema` base DN.

Because the attributes in the schema are operational attributes, you must include `"+"` at the end of your search.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -b cn=schema --searchScope base "(objectclass=*)" "+"
dn: cn=schema
nameForms: ( 1.3.6.1.1.10.15.1 NAME 'uddiBusinessEntityNameForm' OC uddiBusiness
 Entity MUST ( uddiBusinessKey ) X-ORIGIN 'RFC 4403' )
nameForms: ( 1.3.6.1.1.10.15.2 NAME 'uddiContactNameForm' OC uddiContact MUST
```

```
 (uddiUUID ) X-ORIGIN 'RFC 4403' )
nameForms: ( 1.3.6.1.1.10.15.3 NAME 'uddiAddressNameForm' OC uddiAddress MUST
 (uddiUUID ) X-ORIGIN 'RFC 4403' )
...
attributeTypes: ( 1.3.6.1.1.1.1.12 NAME 'memberUid' EQUALITY caseExactIA5Match
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORIGIN 'draft-howard-rfc2307bis' )
attributeTypes: ( 1.3.6.1.1.1.1.13 NAME 'memberNisNetgroup' EQUALITY caseExactIA
 5Match SUBSTR caseExactIA5SubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
 X-ORIGIN 'draft-howard-rfc2307bis' )
attributeTypes: ( 1.3.6.1.1.1.1.14 NAME 'nisNetgroupTriple' DESC 'Netgroup
 triple' EQUALITY caseIgnoreIA5Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORIGIN
 'draft-howard-rfc2307bis' )
...
```

18.5.1.5 Searching the Configuration Entry

The directory server stores its configuration under the `cn=config` entry. Direct access to this entry over LDAP is not advised. The configuration is accessible and modifiable by using the `dsconfig` command. The `dsconfig` command connects to the directory server over SSL through the administration connector. For more information, see Managing Administration Traffic to the Server.

To search the configuration entry using `dsconfig` in interactive mode, run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file
```

For more information about accessing the server configuration by using `dsconfig`, see Managing the Server Configuration Using `dsconfig`.

18.5.1.6 Searching the Monitoring Entry

The directory server monitor entry `cn=monitor` provides statistical information about the server performance, state, and version. You can access this information by using the `ldapsearch` command.

Although you can access `cn=monitor` using any configured LDAP connection handler, it is recommended that you use the administration connector for all access to administrative suffixes. Using the administration connector ensures that monitoring data is not polluted and that server administration takes precedence over user traffic. To use the administration connector, specify the administration port, and include the `--useSSL` option. For more information, see Managing Administration Traffic to the Server.

Run the `ldapsearch` command on the base DN `cn=monitor`.

```
$ ldapsearch -h localhost -p 4444 --useSSL -D "cn=Directory Manager" \
  -j pwd-file -b cn=monitor "(objectclass=*)"
dn: cn=monitor
startTime: 20120119135658Z
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
cn: monitor
vendorName: Oracle Corporation
currentTime: 20120125145650Z
vendorVersion: Oracle Unified Directory 11.1.2.0.0
maxConnections: 3
productName: Oracle Unified Directory
currentConnections: 1
totalConnections: 22
```

```
upTime: 6 days 0 hours 59 minutes 52 seconds
...
```

18.5.2 Searching Over SSL

You can search using client authentication, if you have configured the directory server to accept SSL connections by using a self-signed certificate or certificate.

The following topics describes procedures to show how to search the directory over SSL using various authentication mechanisms:

- Searching Over SSL With Blind Trust
- Searching Over SSL Using a Trust Store
- Searching Over SSL With No Trust Store
- Searching Over SSL Using a Keystore
- Searching Using useStartTLS
- Searching Using SASL With DIGEST-MD5 Client Authentication
- Searching Using SASL With the GSSAPI Mechanism
- Searching Using SASL With the PLAIN Mechanism

18.5.2.1 Searching Over SSL With Blind Trust

You can configure the client to automatically trust any certificate that the server presents to it. However, this method is not secure and is vulnerable to man-in-the-middle attacks. Generally, you should use this type of authentication for testing purposes only.

Run the `ldapsearch` command with the `--trustAll` option.

The following command searches the Root DSE.

```
$ ldapsearch -h localhost -p 1636 --useSSL --trustAll -b "" \
  --searchScope base "(objectClass=*)"
```

18.5.2.2 Searching Over SSL Using a Trust Store

You can configure the client to use a certificate trust store, which contains information about the certificates it can trust. The client can check any server certificate to those listed in its trust store. If the client finds a match, a secure communication can take place with the server. If no match is found, the server cannot be trusted. You must ensure that the presented certificate is valid and add it to the trust store, which then allows secure communication.

Run the `ldapsearch` command with the `--trustStorePath` option.

The following command searches the Root DSE using a trust store.

```
$ ldapsearch -h localhost -p 1636 --useSSL \
  --trustStorePath /home/scarter/security/cert.db -b "" \
  --searchScope base "(objectClass=*)"
```

18.5.2.3 Searching Over SSL With No Trust Store

If no trust store is specified, then you are prompted about whether the certificate that was presented to the client should be trusted.

Run the `ldapsearch` command without the `--trustStorePath` option.

The following command searches the Root DSE without using a trust store.

```
$ ldapsearch -h localhost -p 1636 --useSSL -b "" \
  --searchScope base "(objectclass=*)"
The server is using the following certificate:
Subject DN: CN=example.com, O=Example Corp, C=US
Issuer DN: CN=example.com, O=Example Corp, C=US
Validity: Fri Mar 02 16:48:17 CST 2007 through Thu May 31 17:48:17 CDT 2007
Do you wish to trust this certificate and continue connecting to the server?
Please enter "yes" or "no": yes

dn: objectClass: ds-rootDSE
objectClass: top
```

18.5.2.4 Searching Over SSL Using a Keystore

If the client is required to present its own certificate to the directory server, that client must know which certificate keystore to use. The client can determine the certificate keystore by specifying the `--keyStorePath` option with either the `--keyStorePassword` or `--keyStorePasswordFile`. This scenario typically occurs when the client performs a SASL EXTERNAL authentication or if the server always requires the client to present its own certificates.

Run the `ldapsearch` command with the `--keyStore...` options.

The following command searches the Root DSE using a trust store and a key store.

```
$ ldapsearch -h localhost -p 1636 --useSSL \
  --keyStorePath /home/scarter/security/key.db \
  --keyStorePasswordFile /home/keystore.pin \
  --trustStorePath /home/scarter/security/cert.db --useSASLExternal -b "" \
  --searchScope base "(objectClass=*)"
```

18.5.2.5 Searching Using useStartTLS

The process for using `useStartTLS` with the `ldapsearch` utility is very similar to the process for using SSL. However, you must do the following:

- Use the port on which the server is listening for *unencrypted* LDAP requests
- Indicate that `useStartTLS` should be used instead of SSL (that is, use the `--useStartTLS` option instead of the `--useSSL` option).

Run the `ldapsearch` command with the `--useStartTLS` option.

The following command searches the Root DSE using useStartTLS.

```
$ ldapsearch -h localhost -p 1389 --useStartTLS \
  -b "" --searchScope base "(objectClass=*)"
```

18.5.2.6 Searching Using SASL With DIGEST-MD5 Client Authentication

The directory server supports several Simple Authentication and Security Layer (SASL) mechanisms. DIGEST-MD5 is one form of SASL authentication to the server that does not expose the clear-text password.

Run the `ldapsearch` command with the appropriate `--saslOption` options.

The `authid` option specifies the identity of the user that is authenticating to the server. The option can be in the form of a `dn` (for example, `dn:uid=scarter,dc=example,dc-com`) or a user name (for example, `authid=u:sam.carter`). The attribute can be used to indicate that the search operation should be performed under the authority of another user after authentication. The `realm` specifies the fully qualified name of the server host machine and is optional.

This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1636 --useSSL \
  --trustStorePath /home/cert.db --certNickName "my-cert" -w - \
  --saslOption mech=DIGEST-MD5 --saslOption realm="example.com" \
  --saslOption authid="dn:uid=scarter,dc=example,dc=com" -b "" "(objectclass=*)"
```

18.5.2.7 Searching Using SASL With the GSSAPI Mechanism

The GSSAPI mechanism performs authentication in a Kerberos environment and requires that the client system be configured to participate in such an environment.

Run the `ldapsearch` command to search as a user who already has a valid Kerberos session.

The `authid` attribute specifies the authentication ID that should be used to identify the user.

This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1389 --saslOption mech=GSSAPI \
  --saslOption authid="dn:uid=scarter,dc=example,dc=com" \
  --searchScope "" -b "" "(objectclass=*)"
```

18.5.2.8 Searching Using SASL With the PLAIN Mechanism

The PLAIN mechanism performs authentication in a manner similar to LDAP simple authentication except that the user is identified in the form of an authorization ID rather than a full DN.

Run the `ldapsearch` command to search as a user who already has a valid Kerberos session.

The `authid` attribute specifies the authentication ID that should be used to identify the user.

This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1389 \
  --saslOption mech=PLAIN --saslOption authid="dn:uid=scarter,dc=example,dc=com" \
  --searchScope "" -b "" "(objectclass=*)"
```

18.5.3 Searching Using Controls

LDAP controls extend the functionality of LDAP commands, such as `ldapsearch`, to perform additional operations on top of the search. Each control is defined as an object identifier (OID) that uniquely identifies the control, a criticality flag, and any associated values. If the client sets the criticality flag when sending the control to the directory server, the directory server must either perform the operation with the control or not process it. If the flag is not set by the client, the directory server is free to ignore the control if it cannot process it.

You can use multiple controls in a single operation, such as the virtual list view with server-side sorting. The virtual list view control requires additional explanation and is therefore described in its own section.

This section contains the following topics:

- Viewing the Available Controls

- Searching Using the Join Search Control
- Searching Using the Proximity Search Control
- Searching Using the Account Usability Request Control
- Searching Using the Authorization Identity Request Control
- Searching Using the Get Effective Rights Control
- Searching Using the LDAP Assertion Control
- Searching Using the LDAP Subentry Control
- Searching Using the Manage DSA IT Control
- Searching Using the Matched Values Filter Control
- Searching Using the Password Policy Control
- Searching Using the Persistent Search Control
- Searching Using the Proxied Authorization Control
- Searching Using the Server-Side Sort Control
- Searching Using the Simple Paged Results Control
- Searching Using the Virtual List View Control

18.5.3.1 Viewing the Available Controls

You can view the current list of controls for your directory server by searching the Root DSE entry for the `supportedControl` attribute.

Run the `ldapsearch` command on the Root DSE entry.

```
$ ldapsearch -h localhost -p 1389 -b "" --searchScope base "(objectclass=*)" \
  supportedControl
dn:
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.2.840.113556.1.4.1413
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.840.113556.1.4.473
supportedControl: 1.2.840.113556.1.4.805
supportedControl: 1.3.6.1.1.12
supportedControl: 1.3.6.1.1.13.1
supportedControl: 1.3.6.1.1.13.2
supportedControl: 1.3.6.1.4.1.26027.1.5.2
supportedControl: 1.3.6.1.4.1.26027.1.5.5
supportedControl: 1.3.6.1.4.1.26027.1.5.6
supportedControl: 1.3.6.1.4.1.26027.2.3.1
supportedControl: 1.3.6.1.4.1.26027.2.3.2
supportedControl: 1.3.6.1.4.1.42.2.27.8.5.1
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.2
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.8
supportedControl: 1.3.6.1.4.1.4203.1.10.1
supportedControl: 1.3.6.1.4.1.4203.1.10.2
supportedControl: 2.16.840.1.113730.3.4.12
supportedControl: 2.16.840.1.113730.3.4.16
supportedControl: 2.16.840.1.113730.3.4.17
supportedControl: 2.16.840.1.113730.3.4.18
supportedControl: 2.16.840.1.113730.3.4.19
supportedControl: 2.16.840.1.113730.3.4.2
supportedControl: 2.16.840.1.113730.3.4.3
supportedControl: 2.16.840.1.113730.3.4.4
```

```
supportedControl: 2.16.840.1.113730.3.4.5
supportedControl: 2.16.840.1.113730.3.4.9
supportedControl: 2.16.840.1.113894.1.8.21
supportedControl: 2.16.840.1.113894.1.8.31
```

The controls are returned as a list of OIDs.

> **Note:**
>
> Not all of these controls can be used with the `ldapsearch` command. For a description of the control that corresponds to each, and for more information about supported LDAP controls, see Supported LDAP Controls.

18.5.3.2 Searching Using the Join Search Control

The Join Search Control retrieves related entry tree chains such as friends, managers, and so forth, in a single search operation. The Join Control can only target entry chains with established relationships that can (but do not have to) be cross referenced.

For example, the following entry is part of an established "friends" relationship hierarchy where each participating entry has links to other participating entries. In this case these links are formed by the `friend` attribute.

```
dn: uid=user.3,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: top
uid: user.3
cn: Kenny McCormick
sn: McCormick
friend: uid=user.0,ou=People,dc=example,dc=com
friend: uid=user.1,ou=People,dc=example,dc=com
friend: uid=user.2,ou=People,dc=example,dc=com
...
```

In a search operation with the Join Control, the search parameters such as scope and filter apply to the join search, that is, to entries evaluated during the join. This means that only matching results are returned. This functionality enables you to retrieve the entire linked relationship hierarchy, or a subset of it, in a single search operation, based on specific search criteria and scope.

You can specify the Proximity Search Control with the `ldapsearch` command by using the `--control` or `-J` option with the Proximity Search Control OID (1.3.6.1.4.1.26027.2.3.1) as follows:

```
OID:criticality:attribute
```

where *attribute* is the attribute on which the relationship between entries is based.

The following example requests the subset of user entries that are linked through the `friend` attribute.

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
  --baseDN "uid=user.3,ou=People,dc=example,dc=com" \
  --searchScope sub \
```

```
-J "1.3.6.1.4.1.26027.2.3.1:true:friend" \
"(objectClass=person)"
```

In a join search, the search parameters have the following significance:

- `baseDN`

  The search base is used to specify the precise entry from which to start the join search.

- `searchScope`

  The search scope is used to specify distinct levels of join depth.

  - A search scope of `base` retrieves only direct relationships, for example, direct friends that are specified by the `friend` attribute in the sample entry.
  - A search scope of `one` goes one level deep, retrieving direct friends of direct friends of the sample entry.
  - A search scope of `sub` traverses the entire hierarchy chain no matter how many levels.
  - A search scope of `subordinate` has the same effect as `sub`, but does not include the base entry in search results.

- `filter`

  The search filter is used to evaluate candidate entries during the join for inclusion in the search results. The filter can be used to refine the search to include only specific entries. It works in exactly the same way as the filter for standard search operations but is applicable only to join search results.

18.5.3.3 Searching Using the Proximity Search Control

The Proximity Search Control provides base location data to the server in the search request, which enables the server to generate proximity virtual attribute values for all candidate entries that include location data. The value of the `location` attribute in an entry is the latitude-longitude GPS coordinates, in WGS84 standard format. User applications can periodically update the value of this attribute with the last known location of the user.

For example, the following entry extract shows an entry whose location has been updated to the coordinates of Golden Gate Bridge:

```
dn: uid=user.1,ou=People,dc=example,dc=com
objectClass: geoObject
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: top
objectClass: geoObject
uid: user.1
cn: Bob Smith
sn: Smith
location: 37.81997, -122.47859
...
```

The server can calculate the location proximity of each entry to the base location provided in the Proximity Search Control.

A client application can therefore request a proximity value to be calculated and returned for each matching search result entry. The client application can use the proximity attribute in the search filter of the search operation itself and can therefore request matching search result entries based on their proximity to a given base location.

You can specify the Proximity Search Control with the `ldapsearch` command by using the `--control` or `-J` option with the Proximity Search Control OID (1.3.6.1.4.1.26027.2.3.2) as follows:

```
OID:criticality:location
```

where *location* represents the latitude-longitude GPS coordinates in WGS84 standard format.

The following example sets the base location to the coordinates of the Eiffel Tower (48.858844, 2.294351) and requests all user entries whose location is within 500 meters of the base location.

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
  -b "dc=example,dc=com" --searchScope sub \
  -J "1.3.6.1.4.1.26027.2.3.2:true:48.858844,2.294351" \
  "(&(objectClass=person)(proximity<=500))"
```

18.5.3.4 Searching Using the Account Usability Request Control

The Account Usability Request Control determines if a user account can be used to authenticate to a server. If the user account is available, the control adds a message before any entry about whether the account is usable.

You can specify the Account Usability Request Control with `ldapsearch` in the following ways:

- **OID.** Use the `--control` or `-J` option with the Account Usability Request Control OID: `1.3.6.1.4.1.42.2.27.9.5.8` with no value.
- **Named constant.** Use a named constant, `accountusable` or `accountusability`, with the `--control` or `-J` option, instead of using the Account Usability Request Control OID. For example, use `-J accountusable` or `-J accountusability` with the `ldapsearch` command.

Use the `ldapsearch` command with the `--control` option or its short form `-J`.

```
$ ldapsearch -h localhost -p 1389 -b "dc=example,dc=com" \
  --searchScope sub -J "accountusability:true" "(objectclass=*)"
# Account Usability Response Control
# The account is usable
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
...
```

18.5.3.5 Searching Using the Authorization Identity Request Control

The Authorization Identity Request Control allows the client to obtain the authorization identity for the client connection during the LDAP bind request. The authorization ID returned by the server is displayed to the client as soon as authentication has completed. The line containing the authorization ID is prefixed with a `#` character, making it a comment if the output is to be interpreted as an LDIF.

You can specify the Authorization Identity Request Control with `ldapsearch` in the following ways:

- **OID.** Use the `--control` or `-J` option with the Authorization Identity Request Control OID: `2.16.840.1.113730.3.4.16` with no value.

- **Named constant.** Use a named constant, `authzid` or `authorizationidentity` with the `-control` or `-J` option instead of using the Authorization Identity Request Control OID. For example, use `-J authzid` or `-J authorizationidentity` with the `ldapsearch` command.

Use the `ldapsearch` command with the `--reportAuthzID` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -b dc=example,dc=com --searchScope base \
  --reportAuthzID "(objectclass=*)"
# Bound with authorization ID dn:cn=Directory Manager,cn=Root DNs,cn=config
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

18.5.3.6 Searching Using the Get Effective Rights Control

The Get Effective Rights Control enables you to evaluate existing or new ACIs and to see the effective rights that they grant for a user on a specified entry.

The response to this control is to return the effective rights information about the entries and attributes in the search results. This extra information includes read and write permissions for each entry and for each attribute in each entry. The permissions can be requested for the bind DN used for the search or for an arbitrary DN, allowing administrators to test the permissions of directory users.

The `ldapsearch` command provides two ways to use the Get Effective Rights Control:

- Use `-J effectiverights` or the OID `-J "1.3.6.1.4.1.42.2.27.9.5.2"`. The request only takes an authorization ID (`authzid`). If you specify a NULL value for the authorization ID (`authzid`), the bind user is used as the `authzid`.
- Use `-g dn:"dn"`. The command option shows the effective rights of the user binding with the given DN. You can use this option together with the `-e` option to include the effective rights on the named attributes. You can use the option to determine if a user has permission to add an attribute that does not currently exist in an entry.

> **Note:**
>
> You cannot use the `-g` option with the `-J` option.

To view effective rights, specify the virtual attributes `aclRights` and `aclRightsInfo`, which the server generates in response to the effective rights request. Thus, you should not use these attributes in search commands of any kind.

1. Use the `ldapsearch` command to display the effective rights of all users.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com -J effectiverights "(objectclass=*)" aclRights

dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=People, dc=example,dc=com
```

```
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

...
```

2. Use the `ldapsearch` command to display the effective rights of a specific user.

   This example uses the `--getEffectiveRightsAuthzid` option. You can also use the `--control` or `-J` option, such as `-J geteffectiverights`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com \
  --getEffectiveRightsAuthzid "dn:uid=scarter,ou=People,dc=example,dc=com" \
  "(uid=scarter)" aclRights
dn: uid=scarter,ou=People,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

3. Use the `ldapsearch` command to display effective rights information for a specific user.

   The `aclRightsInfo` attribute provides more detailed logging information that explains how effective rights are granted or denied.

```
ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com \
  --getEffectiveRightsAuthzid "dn:uid=scarter,ou=People,dc=example,dc=com"\
  "(uid=scarter)" aclRightsInfo

dn: uid=scarter,ou=People,dc=example,dc=com
aclRightsInfo;logs;entryLevel;add: acl_summary(main): access not allowed(add) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;proxy: acl_summary(main): access not allowed(proxy )
on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter, ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;write: acl_summary(main): access allowed(write) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: evaluated allow , deciding_aci : Allow self entry
modification)
aclRightsInfo;logs;entryLevel;read: acl_summary(main): access allowed(read) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: evaluated allow , deciding_aci: Anonymous extended
 operation access)
aclRightsInfo;logs;entryLevel;delete: acl_summary(main): access not allowed(delete)
on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
```

18.5.3.7 Searching Using the LDAP Assertion Control

The LDAP Assertion Control allows you to specify a condition that must evaluate to true for the searching operation to process. The value of the control should be in the form of an LDAP

search filter. The server tests the base object before searching for entries that match the search scope and filter. If the assertion fails, no entries are returned.

This example determines first if the assertion is met, and returns the entry if it matches the search filter.

Run the `ldapsearch` command with the `--assertionFilter` option using the assertion `(objectclass=top)`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=HR Managers,ou=Groups,dc=example,dc=com" \ -s sub \
  --assertionFilter "(objectclass=top)" "(objectclass=*)"
dn: cn=HR Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage HR entries
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=cschmith, ou=People, dc=example,dc=com
cn: HR Managers
```

18.5.3.8 Searching Using the LDAP Subentry Control

The LDAP Subentry Control allows the client to request that the server return only entries with the `ldapSubEntry` object class during a search operation. LDAP subentries are *operational objects*, similar to operational attributes, that are returned only if they are explicitly requested. Typically, you can use the control when searching the schema.

You request the server to return subentries with `ldapsearch` in the following ways:

- Using the `--subEntries` option to specify the LDAP Subentry Control.
- Specifying base search scope to retrieve a specific subentry if its base DN is known.
- Using the equality filter, `(objectclass=ldapSubentry)`.

> **Note:**
>
> Using the equality filter is not part of the standard and is supported for backward compatibility only.

Run the `ldapsearch` command with the `--subEntries` option, as follows:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=schema" --subEntries "(objectclass=*)"
```

18.5.3.9 Searching Using the Manage DSA IT Control

The Manage DSA IT Control allows the client to request that the server treat smart referrals as regular entries during the search. A *smart referral* is an entry that references another server or location in the directory information tree `DIT` and contains the `referral` object class with one or more attributes containing the LDAP URLs that specify the referral.

You can specify the Manage DSA IT Control with `ldapsearch` in the following ways:

- **OID**. Use the `--control` or `-J` option with the Manage DSA IT Control OID: `2.16.840.1.113730.3.4.2` with no value.

- **Named constant**. Use the named constant, `managedsait` with the `--control` or `-J` option instead of the Manage DSA IT Control OID. For example, use `-J managedsait` with the `ldapsearch` command.

To use the Manage DSA IT control in a search, run the `ldapsearch` command with the `-J` option, as follows:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com -J managedsait "(uid=president)" ref
dn: uid=president,ou=People,dc=example,dc=com
ref: ldap://example.com:389/dc=example,dc=com??sub?(uid=bjensen)
```

> **Note:**
>
> Without the `-J managedsait` argument, the command returns the referred entry.

18.5.3.10 Searching Using the Matched Values Filter Control

The Matched Values Filter Control allows clients to request a subset of attribute values from an entry that evaluate to TRUE. This control allows the user to selectively read a subset of attribute values without retrieving all values, and then scan for the desired set locally.

Run the `ldapsearch` command with the `--matchedValuesFilter` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b ou=groups,dc=example,dc=com --matchedValuesFilter "(uniquemember=uid=kvaughan*)"
  "(objectclass=*)"
dn: ou=Groups,dc=example,dc=com
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
dn: cn=HR Managers,ou=groups,dc=example,dc=com
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
dn: cn=QA Managers,ou=groups,dc=example,dc=com
dn: cn=PD Managers,ou=groups,dc=example,dc=com
```

18.5.3.11 Searching Using the Password Policy Control

The Password Policy Control allows a client to request information about the current password policy information for a user entry.

You can specify the Password Policy Control with `ldapsearch` in the following ways:

- **OID**. Use the `--control` or `-J` option with the Password Policy Control OID: `1.3.6.1.4.1.42.2.27.8.5.1` with no value.
- **Named constant**. Use the named constants, `pwpolicy` or `passwordpolicy` with the `--control` or `-J` option instead of the Password Policy Control OID. For example, use `-J pwpolicy` or `-J passwordpolicy` with `ldapsearch`.
- **Option**. Use the `--usePasswordPolicyControl` option.

> **Note:**
>
> The `-J` or `--control` option is used to specify which controls to use in a *search* request. The `--usePasswordPolicyControl` option is used for *bind* requests.

Run the `ldapsearch` command with the `--usePasswordPolicyControl` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com -s base --usePasswordPolicyControl "(objectclass=*)"
```

18.5.3.12 Searching Using the Persistent Search Control

The Persistent Search Control allows a client to receive notification when entries in the directory are changed by an add, delete, or modify operation. When a change occurs, the server sends the updated entry to the client if the entry matches the search criteria that was used by the Entry Change Notification Control.

The `ldapsearch` command provides an option to run a persistent search (`-C`) that keeps the connection open and displays the entries that match the scope and filter whenever any changes (add, delete, modify, or all) occur. You can quit the search by pressing `Control-C`.

The value for this argument must be in the form:
`ps[[:''changetype''[[:''changesonly''[[:''entrychangecontrols'']]]]`

The elements of this value include the following:

- `ps` — Required operator.
- `changetype` — Indicates the types of changes for which the client wants to receive notification. This element can be any of `add`, `del`, `mod`, or `moddn`, or it can be `all` to register for all change types. It can also be a comma-separated list to register for multiple specific change types. If this element is not provided, it defaults to including `all` change types.
- `changesonly` — If `True`, the client should only be notified of changes that occur to matching entries after the search is registered. If `False`, the server should also send all existing entries in the server that match the provided search criteria. If this element is not provided, then it will default to only returning entries for updates that have occurred since the search was registered.
- `entrychangecontrols` — If `True`, the server should include the Entry Change Notification Control in entries sent to the client as a result of changes. If `False`, the Entry Change Notification Control should not be included. If this element is not provided, then it will default to including the Entry Change Notification Controls.

1. Run the `ldapsearch` command as follows:

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=admin,cn=Administrators,cn=config" \
     -j pwd-file -b dc=example,dc=com --persistentSearch ps:add:true:true \
     "(objectclass=*)"
   ```

   > **Note:**
   >
   > When you use this command, the server waits for any changes made using `add`, `delete`, `modify` or `all` to return values.

2. Open another terminal window and use `ldapmodify` to add a new entry.

```
$ ldapmodify -h localhost -p 1389 -b dc=example,dc=com \
  --defaultAdd --filename new_add.ldif
Processing ADD request for uid=Marcia Garza,ou=People,dc=example,dc=com
ADD operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

3. The original terminal window shows the change.

   To end the session, press Control-Z (UNIX/Linux) or Control-C (Windows).

```
# Persistent search change type:  add
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenName: Marcia
uid: mgarza
uid: Marcia Garza
cn: Marcia Garza
sn: Garza
userpassword: {SSHA}SNfL1RUm5uvTnLK+G0K3oz+Peb1i5/+YsylfBg==
roomnumber: 5484
l: Santa Clara
ou: Accounting
ou: People
mail: mgarza@example.com
```

4. To terminate the session, press Control-D (UNIX/Linux) or Control-C (Windows), and then type `Y` to quit.

```
Terminate batch job (Y/N)?
```

18.5.3.13 Searching Using the Proxied Authorization Control

The Proxied Authorization Control allows a client to impersonate another entry for a specific operation. This control can be useful in trusted applications that need to perform on behalf of many different users, so that the application does not need to re-authenticate for each operation.

Run the `ldapsearch` command with the `--proxyAs` option, as follows:

Here, `clientApp` must have the appropriate ACI permissions within the subtree to use the Proxied Authorization Control. If not granted, LDAP error 50 `insufficient access rights` will be returned to the client.

```
$ ldapsearch -h localhost -p 1389 \
  -D "uid=clientApp,ou=Applications,dc=example,dc=com" -j pwd-file \
  -s sub -b dc=example,dc=com \
  --proxyAs "dn:uid=acctgAdmin,ou=Administrators,ou=People,dc=example,dc=com" \
  "(uid=kvaughan)" mail
```

18.5.3.14 Searching Using the Server-Side Sort Control

The Server-Side Sort Control allows the client to request that the server sort the search results before sending them to the client. This is convenient when the server has indexes that can satisfy the sort order requested by the client faster than the client can.

You can sort the number of entries returned by using the `--sortOrder` option. If you do not specify `+` (a plus sign) for ascending or `-` (a minus sign) for descending, then the default option is to sort in ascending order.

1. Use the `ldapsearch` command to search all entries and to display the results in ascending order.

   Use the `--sortOrder` option sorted on the attributes `sn` and `givenName`.

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --s sub -b dc=example,dc=com --sortorder sn,givenName "(objectclass)"
   dn: uid=dakers,ou=People,dc=example,dc=com
   objectClass: person
   objectClass: organizationalPerson
   ...<search results>...
   ```

2. Use the `ldapsearch` command to search all entries and display the results in descending order.

   Use the `--sortorder` option sorted on the attribute `sn`.

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     -s sub -b dc=example,dc=com --sortOrder -sn "(objectclass)"
   dn: uid=pworrell,ou=People,dc=example,dc=com
   objectClass: person
   objectClass: organizationalPerson
   ...<search results>...
   ```

18.5.3.15 Searching Using the Simple Paged Results Control

The Simple Paged Results Control allows a search operation to return only a subset of the results at a time. It can be used to iterate through the search results a page at a time. It is similar to the Virtual List View Control except that it does not require the results to be sorted and can only be used to iterate sequentially through the search results.

Use the `ldapsearch` command with the `--simplePageSize` option.

The following command also uses the `--countEntries` option to mark each page.

```
$ ldapsearch --hostname localhost --port 1389 \
 --bindDN "cn=Directory Manager" --bindPassword password \
 --searchScope sub --baseDN dc=example,dc=com \
 --simplePageSize 2 --countEntries "(objectclass=*)"

dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups

dn: ou=People,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: People

# Total number of matching entries: 2

dn: ou=Special Users,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
description: Special Administrative Accounts
ou: Special Users

dn: ou=Company Servers,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
description: Standard branch for Company Server registration
ou: Company Servers
```

```
# Total number of matching entries: 2

dn: ou=Contractors,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
ou: Contractors
ou: Product Testing
ou: Product Development
ou: Accounting

# Total number of matching entries: 1
```

18.5.3.16 Searching Using the Virtual List View Control

The following topics describe procedures to search using the Virtual List View control:

- About the Virtual List View Control
- Searching Using the Virtual List View Control
- Searching Using Virtual List View With a Specific Target
- Searching Using Virtual List View With a Known Total
- Allowing Anonymous Access to the Virtual List View Control

18.5.3.16.1 About the Virtual List View Control

The Virtual List View Control allows a client to request that the server send search results in small, manageable chunks within a specific range of entries. It also allows a client to move forward and backward through the results of a search operation if configured with a GUI browser or application, or jump directly to a particular entry.

> **Note:**
>
> The Virtual List View Control requires that the returned entries be sorted.

Together with the `--virtualListView` option or its short form `-G`, specify the following arguments:

- **before**. Specify the number of entries before the target to include in the results.

  If the `before` value is greater than or equal to the target offset, then the `before` value is adjusted so that the first entry returned is the beginning of the list.
- **after**. Specify the number of entries after the target to include in the results.
- **index**. Specify the offset of the target entry within the result set. An index of 1 always means the first entry. If `index` and `content_count` are equal, the last entry is selected.

  If the `index` value is negative, the server rejects the request.

  If the `index` value is `0`, it is adjusted to `1` so that returned values are displayed.

  If the `index` value is greater than the total number of matching values, it is adjusted to one greater than the content count.

The value of `index` can also be an assertion value, so that the returned entry contains that value. If the returned entry is so near the end of the list that the value of `after` extends beyond the last entry, the value of `after` is adjusted to display the appropriate entries.

- **count**. Specify the expected size of the result set.
  - **count=0**. The target entry is the entry at the specified *index* position, starting from 1 and relative to the entire list of sorted results. Use this argument if the client does not know the size of the result set.
  - **count=1**. The target entry is the first entry in the list of sorted results.
  - **count>1**. The target entry is the first entry in the portion of the list represented by the fraction *index/count*. To target the last result in the list, use an *index* argument greater than the *count* argument. Client applications can use interfaces that allow users to move around a long list by using a scroll bar. For example, for an index of 33 and a count of 100, the application can jump 33 percent of the way into the list.

For example, the arguments (0:4:1:0) indicate that you want to show 0 entries before and 4 entries after the target entry at index 1. If the client does not know the size of the set, the count is 0.

18.5.3.16.2 Searching Using the Virtual List View Control

The sort order option (`-S`) must be used with the Virtual List View control. This example uses the Virtual List View Control options to specify the following:

- **Before=0.** Specifies that 0 entries before the target should be displayed.
- **After=2.** Specifies that 2 entries after the target should be displayed.
- **Index=1.** Specifies that the offset of the target entry within the result set should be returned.
- **Count=0.** Specifies that target entry at the index position should be returned, which is the first entry.

Thus, the server returns the first entry plus two entries after the target sorted in ascending order by the `givenName` attribute.

Use the `ldapsearch` command with the `--virtualListView` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w bindPassword \
  -b dc=example,dc=com --searchScope sub --sortOrder givenName \
  --virtualListView "0:2:1:0" "(objectclass=*)"

dn: uid=awhite,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alan
uid: awhite
cn: Alan White
sn: White
...

dn: uid=aworrell,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alan
```

```
uid: aworrell
cn: Alan Worrell
sn: Worrell
...

dn: uid=alutz,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alexander
uid: alutz
cn: Alexander Lutz
sn: Lutz
...

# VLV Target Offset: 1
# VLV Content Count: 172
```

18.5.3.16.3 Searching Using Virtual List View With a Specific Target

You can use the sort order (`-S`) option with Virtual List View.

The example command uses the Virtual List View Control options to specify the following:

- **Before=0.** Specifies that 0 entries before the target should be displayed.
- **After=4.** Specifies that 4 entries after the target should be displayed.
- **Index=jensen.** Specifies that the string `jensen` within the result set be returned.
- **Count=not specified.** Use the default `count=0`, which is the first entry.

Thus, the server returns the first `sn` attribute that matches `jensen` plus four `sn` attributes after the target sorted in ascending order by the `sn` attribute.

Use the `ldapsearch` command with the `--virtualListView` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com --searchScope sub --sortOrder sn \
  --virtualListView "0:4:jensen" "(objectclass=*)" sn

dn: uid=kjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=bjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=gjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=jjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=ajensen,ou=People,dc=example,dc=com
sn: Jensen

# VLV Target Offset: 56
# VLV Content Count: 172
```

18.5.3.16.4 Searching Using Virtual List View With a Known Total

The sort order (`-S`) option must also be used with Virtual List View.

The example command uses the Virtual List View Control options to specify the following:

- **Before=0.** Specifies that 0 entries before the target should be displayed.
- **After=2.** Specifies that 2 entries after the target should be displayed.
- **Index=57.** Specifies that the index of 57 within the result set should be returned. This is roughly one-third of the list.
- **Count=172.** Use the total count.

Thus, the server returns the first `sn` attribute that is one-third within the list, plus two `sn` attributes sorted in ascending order by the `sn` attribute.

Use the `ldapsearch` command with the `--virtualListView` option.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com -s sub --sortOrder sn \
  --virtualListView "0:2:57:172" "(objectclass=*)" sn

dn: uid=bjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=gjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=jjensen,ou=People,dc=example,dc=com
sn: Jensen

# VLV Target Offset: 57
# VLV Content Count: 172
```

18.5.3.16.5 Allowing Anonymous Access to the Virtual List View Control

By default, access to the virtual list view control is allowed for authenticated users only. To allow unauthenticated users to access the virtual list view control, the OID for the virtual list view control (2.16.840.1.113730.3.4.9) must be added to the "Anonymous control access" global ACI and removed from the "Authenticated users control access" global ACI.

```
ds-cfg-global-aci: (targetcontrol="2.16.840.1.113730.3.4.2 ||
2.16.840.1.113730.3.4.17 || 2.16.840.1.113730.3.4.19 ||
1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 ||
2.16.840.1.113730.3.4.16 || 2.16.840.1.113894.1.8.31") (version 3.0; acl
"Anonymous control access"; allow(read) userdn="ldap:///anyone";)
ds-cfg-global-aci: (targetcontrol="1.3.6.1.1.12 || 1.3.6.1.1.13.1 ||
1.3.6.1.1.13.2 || 1.2.840.113556.1.4.319 || 1.2.826.0.1.3344810.2.3 ||
2.16.840.1.113730.3.4.18 || 2.16.840.1.113730.3.4.9 || 1.2.840.113556.1.4.473
|| 1.3.6.1.4.1.42.2.27.9.5.9 || 1.2.840.113556.1.4.473") (version 3.0; acl
"Authenticated users control access"; allow(read) userdn="ldap:///all";)
```

The easiest way to modify these global ACIs is to use OUDSM, as follows:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Under the **Root** menu, select **Anonymous control access**.
4. In the **Targets** table on the right hand pane, select the **Target Control** field, and click **Edit**.

5. From the **Available Controls** list, select **Virtual List View Control (2.16.840.1.113730.3.4.9)**.
6. Click the right arrow to move the VLV control to the **Selected Controls** list.
7. Click **OK**.
8. Click **Apply** to save your changes.
9. Under the **Root** menu, select **Authenticated users control access**.
10. In the **Targets** table on the right hand pane, select the **Target Control** field, and click **Edit**.
11. From the **Selected Controls** list, select **Virtual List View Control (2.16.840.1.113730.3.4.9)**.
12. Click the left arrow to move the VLV control to the **Available Controls** list.
13. Click **OK**.
14. Click **Apply** to save your changes.

You can also use `dsconfig` to modify the global ACIs, but it is not possible to modify an ACI value with dsconfig. Instead, the ACIs must be deleted and recreated. For more information, see About Default Global ACIs.

18.5.4 Searching in Verbose Mode and With a Properties File

You can understand how to search in verbose mode and by using a properties file. The verbose mode is convenient for debugging purposes whereas the properties file is convenient when working in different configuration environments.

This section contains the following topics:

- Searching in Verbose Mode
- Searching Using a Properties File

18.5.4.1 Searching in Verbose Mode

Verbose mode displays the processing information that is transmitted between client and server. This mode is convenient for debugging purposes.

Use the `ldapsearch` command as follows:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
   -b dc=example,dc=com -s base --verbose "(objectclass=*)"
LDAP: C>S 01:43:46.140 (0ms) LDAPMessage(msgID=1, protocolOp=BindRequest
  (version =3, dn=cn=Directory Manager, password=password))
ASN1: C>S 01:43:46.140 (0ms) ASN.1 Sequence
  BER Type: 30
  Decoded Values:
  ASN1Integer(type=02, value=1)
  ASN1Sequence(type=60, values={ ASN1Integer(type=02, value=3),
    cn=Directory Manager, opends })
  Value:
  02 01 01 60 23 02 01 03   04 14 63 6E 3D 64 69 72     `#  cn=directory
  65 63 74 6F 72 79 20 6D   61 6E 61 67 65 72 80 08          manager
  70 61 73 73 77 6F 72 64                                    password
...
```

18.5.4.2 Searching Using a Properties File

The directory server supports the use of a properties file that holds default argument values used with the `ldapsearch` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

1. Create a properties file in any text editor, with the following content:

```
hostname=localhost
port=1389
bindDN=cn=Directory Manager
bindPasswordFile=pwd-file
baseDN=dc=example,dc=com
searchScope=sub
sortOrder=givenName
virtualListView=0:2:1:0
```

2. Save the file as `tools.properties`.
3. Use the `ldapsearch` with the `--propertiesFilePath` option.

```
$ ldapsearch --propertiesFilePath tools.properties "(objectclass=*)"
```

18.5.5 Searching Internationalized Entries

You can search for Internationalized entries by using collation rules, by understanding search examples and by using supported collation rules.

This section contains the following topics:

- Using Collation Rules to Search Internationalized Entries
- Understanding Search Examples
- Supported Collation Rules

18.5.5.1 Using Collation Rules to Search Internationalized Entries

The following topics describe how to use collation rules to search internationalized entries:

- About Collation Rules
- Examples of Using Collation Rules

18.5.5.1.1 About Collation Rules

Oracle Unified Directory supports collation rules that match entries and can be used with the Searching Using the Server-Side Sort Control to sort search results. The collation rule is specified in the search filter as a matching rule, delimited by colons, as shown here:

locale.*matchingRule*

where:

- *locale* is specified in one of the following ways
  - Locale OID
  - Locale character suffix (such as `ar`, `en`, or `fr-CA`).

    See Supported Collation Rules for a list of supported locales, their OIDs, and tags.

- *matchingRule* can specified as either a numeric suffix or a character suffix appended to the *locale*, as listed in Table 18-1.

> **Note:**
>
> If the locale is specified by its OID, then the matching rule must be specified by its numeric suffix. In this case, the matching rule cannot be specified by the character suffix.

Table 18-1 Matching Rule Suffixes

Matching Rule	Numeric Suffix	Character Suffix
Less than	.1	.lt
Less than or equal to	.2	.lte
Equality	.3	.eq (default)
Greater than or equal to	.4	.gte
Greater than	.5	.gt
Substring	.6	.sub

`Equality` is the default matching rule. That is, when no matching rule suffix is specified, the collation rule uses `equality` matching rule.

18.5.5.1.2 Examples of Using Collation Rules

The following examples are equivalent and specify the English collation rule and the equality matching rule, but the second example specifies the equality matching rule explicitly with the.`eq` suffix:

```
"cn:en:=sanchez"
"cn:en.eq:=sanchez"
```

The next example shows the same search filter, but specified using the locale's character suffix and the matching rule's numeric code:

```
"cn:en.3:=sanchez"
```

The following example shows the same search filter specified using the locale OID and the matching rule numeric suffix:

```
"cn:1.3.6.1.4.1.42.2.27.9.4.34.1.3:=sanchez"
```

The following examples specify the same search filter but with a Spanish collation rule.

```
"cn:es.eq:=sanchez"
"cn:1.3.6.1.4.1.42.2.27.9.4.49.1.3:=sanchez"
"cn:es.3:=sanchez"
```

The following examples specify a similar search filter that uses a greater-than matching rule with the Spanish collation rule.

```
"cn:es.gt:=sanchez"
"cn:1.3.6.1.4.1.42.2.27.9.4.49.1.5:=sanchez"
"cn:es.5:=sanchez"
```

18.5.5.2 Understanding Search Examples

This section describes the various searches available:

- Equality Search

  The following search uses a filter with the `en` (`en-US`) locale OID to perform an equality search to return any entry with a `cn` value of `sanchez`:

  ```
  $ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
    "cn:1.3.6.1.4.1.42.2.27.9.4.34.1:=sanchez"
  ```

  The following filters return the same results:

  - `"cn:en:=sanchez"`
  - `"cn:en.3:=sanchez"`
  - `"cn:en.eq:=sanchez"`
  - `"cn:1.3.6.1.4.1.42.2.27.9.4.34.1.3:=sanchez"`

- Less-Than Search

  The following search uses a filter with the `es` (`es-ES`) locale and performs a less-than search and returns the entry with a `departmentnumber` value of `abc119`:

  ```
  $ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
    "departmentnumber:1.3.6.1.4.1.42.2.27.9.4.49.1.1:=abc120"
  ```

  The following filters return the same results:

  - `"departmentnumber:es.1:=abc120"`
  - `"departmentnumber:es.lt:=abc120"`

- Less-Than-or-Equal-To Search

  The following search uses a filter with the `es` (`es-ES`) locale and performs a less-than-or-equal-to search that returns the entry with a `departmentnumber` value of `abc119`:

  ```
  $ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
    "departmentnumber:1.3.6.1.4.1.42.2.27.9.4.49.1.2:=abc119"
  ```

  The following filters return the same results:

  - `"departmentnumber:es.2:=abc119"`
  - `"departmentnumber:es.lte:=abc119"`

- Greater-Than-or-Equal-To Search

  The following search uses a filter with the `fr` (`fr-FR`) locale and performs a greater-than-or-equal-To search that returns an entry with a `departmentnumber` value of `abc119`

  ```
  $ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
    "departmentnumber:fr.4:=abc119"
  ```

  The following filters return the same results:

  - `"departmentnumber:1.3.6.1.4.1.42.2.27.9.4.76.1.4:=abc119"`
  - `"departmentnumber:fr.gte:=abc119"`

- Greater-Than Search

The following search uses a filter with the `fr` (`fr-FR`) locale and performs a greater-than search:

```
$ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
  "departmentnumber:fr.5:=abc119"
```

The above search should *not* return an entry with a `departmentnumber` value of `abc119`.

The following filters return the same results:

- `"departmentnumber:1.3.6.1.4.1.42.2.27.9.4.76.1.5:=abc119"`
- `"departmentnumber:fr.gt:=abc119"`

- Substring Search

  The following search uses a filter with the `en` (`en-US`) locale and performs a substring search that returns an entry with an `sn` value of "Quebec":

  ```
  $ ldapsearch -D "cn=directory manager" -j pwd-file -b "o=test" \
    "sn:en.6:=*u*bec"
  ```

  The following filters return the same results:

  - `"sn:1.3.6.1.4.1.42.2.27.9.4.34.1.6:=*u*bec"`
  - `"sn:en.sub:=*u*bec"`

18.5.5.3 Supported Collation Rules

The following table lists the internationalization locales supported by Oracle Unified Directory, alphabetized by character suffix.

Table 18-2 Supported Collation Rules

Locale	Character Suffix	OID
Arabic	ar	1.3.6.1.4.1.42.2.27.9.4.3.1
Arabic United Arab Emirates	ar-AE	1.3.6.1.4.1.42.2.27.9.4.4.1
Arabic Bahrain	ar-BH	1.3.6.1.4.1.42.2.27.9.4.5.1
Arabic Algeria	ar-DZ	1.3.6.1.4.1.42.2.27.9.4.6.1
Arabic Egypt	ar-EG	1.3.6.1.4.1.42.2.27.9.4.7.1
Arabic India	ar-IQ	1.3.6.1.4.1.42.2.27.9.4.9.1
Arabic Jordan	ar-JO	1.3.6.1.4.1.42.2.27.9.4.10.1
Arabic Kuwait	ar-KW	1.3.6.1.4.1.42.2.27.9.4.11.1
Arabic Lebanon	ar-LB	1.3.6.1.4.1.42.2.27.9.4.12.1
Arabic Lybia	ar-LY	1.3.6.1.4.1.42.2.27.9.4.13.1
Arabic Morocco	ar-MA	1.3.6.1.4.1.42.2.27.9.4.14.1
Arabic Oman	ar-OM	1.3.6.1.4.1.42.2.27.9.4.15.1
Arabic Qatar	ar-QA	1.3.6.1.4.1.42.2.27.9.4.16.1
Arabic Saudi Arabia	ar-SA	1.3.6.1.4.1.42.2.27.9.4.17.1
Arabic Sudan	ar-SD	1.3.6.1.4.1.42.2.27.9.4.18.1
Arabic Syria	ar-SY	1.3.6.1.4.1.42.2.27.9.4.19.1
Arabic Tunisia	ar-TN	1.3.6.1.4.1.42.2.27.9.4.20.1

Table 18-2 (Cont.) Supported Collation Rules

Locale	Character Suffix	OID
Arabic Yemen	ar-YE	1.3.6.1.4.1.42.2.27.9.4.21.1
Byelorussian	be	1.3.6.1.4.1.42.2.27.9.4.22.1
Bulgaria	bg	1.3.6.1.4.1.42.2.27.9.4.23.1
Catalan	ca	1.3.6.1.4.1.42.2.27.9.4.25.1
Czech	cs	1.3.6.1.4.1.42.2.27.9.4.26.1
Danish	da	1.3.6.1.4.1.42.2.27.9.4.27.1
German	de	1.3.6.1.4.1.142.2.27.9.4.28.1
German Germany	de-DE	1.3.6.1.4.1.142.2.27.9.4.28.1
German Austria	de-AT	1.3.6.1.4.1.42.2.27.9.4.29.1
German Swiss	de-CH	1.3.6.1.4.1.42.2.27.9.4.31.1
German Luxembourg	de-LU	1.3.6.1.4.1.42.2.27.9.4.32.1
Greek	el	1.3.6.1.4.1.42.2.27.9.4.33.1
English	en	1.3.6.1.4.1.42.2.27.9.4.34.1
English US	en-US	1.3.6.1.4.1.42.2.27.9.4.34.1
English Australia	en-AU	1.3.6.1.4.1.42.2.27.9.4.35.1
English Canada	en-CA	1.3.6.1.4.1.42.2.27.9.4.36.1
English Great Britain	en-GB	1.3.6.1.4.1.42.2.27.9.4.37.1
English Ireland	en-IE	1.3.6.1.4.1.42.2.27.9.4.39.1
English India	en-IN	1.3.6.1.4.1.42.2.27.9.4.40.1
English New Zealand	en-NZ	1.3.6.1.4.1.42.2.27.9.4.42.1
English South Africa	en-ZA	1.3.6.1.4.1.42.2.27.9.4.46.1
Spanish	es	1.3.6.1.4.1.42.2.27.9.4.49.1
Spanish Spain	es-ES	1.3.6.1.4.1.42.2.27.9.4.49.1
Spanish Argentina	es-AR	1.3.6.1.4.1.42.2.27.9.4.50.1
Spanish Bolivia	es-BO	1.3.6.1.4.1.42.2.27.9.4.51.1
Spanish Chile	es-CL	1.3.6.1.4.1.42.2.27.9.4.52.1
Spanish Colombia	es-CO	1.3.6.1.4.1.42.2.27.9.4.53.1
Spanish Costa Rica	es-CR	1.3.6.1.4.1.42.2.27.9.4.54.1
Spanish Dominican Republic	es-DO	1.3.6.1.4.1.42.2.27.9.4.55.1
Spanish Ecuador	es-EC	1.3.6.1.4.1.42.2.27.9.4.56.1
Spanish Guatemala	es-GT	1.3.6.1.4.1.42.2.27.9.4.57.1
Spanish Honduras	es-HN	1.3.6.1.4.1.42.2.27.9.4.58.1
Spanish Mexico	es-MX	1.3.6.1.4.1.42.2.27.9.4.59.1
Spanish Nicaragua	es-NI	1.3.6.1.4.1.42.2.27.9.4.60.1
Spanish Panama	es-PA	1.3.6.1.4.1.42.2.27.9.4.61.1
Spanish Peru	es-PE	1.3.6.1.4.1.42.2.27.9.4.62.1
Spanish Puerto Rico	es-PR	1.3.6.1.4.1.42.2.27.9.4.63.1

Table 18-2 (Cont.) Supported Collation Rules

Locale	Character Suffix	OID
Spanish Paraguay	es-PY	1.3.6.1.4.1.42.2.27.9.4.64.1
Spanish Salvador	es-SV	1.3.6.1.4.1.42.2.27.9.4.65.1
Spanish Uraguay	es-UY	1.3.6.1.4.1.42.2.27.9.4.67.1
Spanish Venezuela	es-VE	1.3.6.1.4.1.42.2.27.9.4.68.1
Estonian	et	1.3.6.1.4.1.42.2.27.9.4.69.1
Finnish	fi	1.3.6.1.4.1.42.2.27.9.4.74.1
French	fr	1.3.6.1.4.1.42.2.27.9.4.76.1
French	fr-FR	1.3.6.1.4.1.42.2.27.9.4.76.1
French	fr-BE	1.3.6.1.4.1.42.2.27.9.4.77.1
French	fr-CA	1.3.6.1.4.1.42.2.27.9.4.78.1
French	fr-CH	1.3.6.1.4.1.42.2.27.9.4.79.1
French	fr-LU	1.3.6.1.4.1.42.2.27.9.4.80.1
Hebrew	he	1.3.6.1.4.1.42.2.27.9.4.85.1
Croatian	hr	1.3.6.1.4.1.42.2.27.9.4.87.1
Hungarian	hu	1.3.6.1.4.1.42.2.27.9.4.88.1
Icelandic	is	1.3.6.1.4.1.42.2.27.9.4.91.1
Italian	it	1.3.6.1.4.1.42.2.27.9.4.92.1
Italian-Swiss	it-CH	1.3.6.1.4.1.42.2.27.9.4.93.1
Japanese	ja	1.3.6.1.4.1.42.2.27.9.4.94.1
Korean	ko	1.3.6.1.4.1.42.2.27.9.4.97.1
Lithuanian	lt	1.3.6.1.4.1.42.2.27.9.4.100.1
Latvian	lv	1.3.6.1.4.1.42.2.27.9.4.101.1
Macedonian	mk	1.3.6.1.4.1.42.2.27.9.4.102.1
Dutch	nl	1.3.6.1.4.1.42.2.27.9.4.105.1
Dutch Netherlands	nl-NL	1.3.6.1.4.1.42.2.27.9.4.105.1
Dutch Belgium	nl-BE	1.3.6.1.4.1.42.2.27.9.4.106.1
Norwegian	no	1.3.6.1.4.1.42.2.27.9.4.107.1
Norwegian Norway	no-NO	1.3.6.1.4.1.42.2.27.9.4.107.1
Norwegian Nynorsk	no-NO-NY	1.3.6.1.4.1.42.2.27.9.4.108.1
Polish	pl	1.3.6.1.4.1.42.2.27.9.4.114.1
Portuguese	pt	1.3.6.1.4.1.42.2.27.9.4.115.1
Portuguese Portugal	pt-PT	1.3.6.1.4.1.42.2.27.9.4.115.1
Portugues Brazil	pt-BR	1.3.6.1.4.1.42.2.27.9.4.116.1
Romanian	ro	1.3.6.1.4.1.42.2.27.9.4.117.1
Russian	ru	1.3.6.1.4.1.42.2.27.9.4.118.1
Russian Russia	ru-RU	1.3.6.1.4.1.42.2.27.9.4.118.1
Slovak	sk	1.3.6.1.4.1.42.2.27.9.4.121.1

Table 18-2 (Cont.) Supported Collation Rules

Locale	Character Suffix	OID
Slovenia	sl	1.3.6.1.4.1.42.2.27.9.4.122.1
Albanian	sq	1.3.6.1.4.1.42.2.27.9.4.127.1
Serbian	sr	1.3.6.1.4.1.42.2.27.9.4.128.1
Swedish	sv	1.3.6.1.4.1.42.2.27.9.4.129.1
Swedish Sweden	sv-SE	1.3.6.1.4.1.42.2.27.9.4.129.1
Thai	th	1.3.6.1.4.1.42.2.27.9.4.136.1
Turkish	tr	1.3.6.1.4.1.42.2.27.9.4.140.1
Ukrainian	uk	1.3.6.1.4.1.42.2.27.9.4.141.1
Vietnamese	vi	1.3.6.1.4.1.42.2.27.9.4.142.1
Chinese	zh	1.3.6.1.4.1.42.2.27.9.4.143.1
Chinese China	zh-CN	1.3.6.1.4.1.42.2.27.9.4.144.1
Chinese Hong Kong	zh-HK	1.3.6.1.4.1.42.2.27.9.4.145.1
Chinese Taiwan	zh-TW	1.3.6.1.4.1.42.2.27.9.4.148.1

18.5.6 Sorting Multi-Valued Attributes in a Search Response

Multi-valued attributes in an entry are usually returned in the order they were added or modified. To return them in sorted order, set the `sort-multivalued-attributes` configuration parameter of the DB Local Backend Workflow Element to `true`. By default, this value is `false`.

> **Note:**
>
> You should enable this configuration only for specific user backends where sorting of multi-valued attributes is highly desirable, as it may have an impact on performance. Only User Attributes with an ordering matching rule are considered for sorting. Sorting excludes DNs, operational attributes, binary attributes, objectclass attributes, and groups. This feature is not supported by OUD Proxy.

To enable sorting of multi-valued attributes for a specific backend, set the `sort-multivalued-attributes` parameter to `true` using the `dsconfig` command as follows:

```
dsconfig set-workflow-element-prop \
          --element-name userRoot \
          --set sort-multivalued-attributes:true \
          --hostname hostname \
          --port admin-port \
          --portProtocol LDAP \
       --trustStorePath INSTALL_PATH/asinst_1/OUD/config/admin-truststore \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile ****** \
          --no-prompt
```

Consider the following scenario in which a user has multiple `uid` and `mail` values, without sorting enabled:

```
dn: uid=user1,ou=People,dc=example,dc=com
mail: abc@mycompany
mail: zxy@mycompany
mail: mno@mycompany
sn: user1
cn: user1
objectClass: top
objectClass: organizationalPerson
objectClass: person
objectClass: inetorgperson
uid: user1@mycompany.com
uid: user1
```

When you enable the `sort-multivalued-attributes` configuration parameter and then run `ldapsearch`, it will return the `uid` and `mail` values sorted according to the ordering matching rule.

```
./ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file -b
"dc=example,dc=com" -s sub "(uid=user1)"
dn: uid=user1,ou=People,dc=example,dc=com
mail: abc@mycompany.com
mail: mno@mycompany.com
mail: zxy@mycompany.com
sn: user1
cn: user1
objectClass: top
objectClass: organizationalPerson
objectClass: person
objectClass: inetorgperson
uid: user1
uid: user1@mycompany.com
```

> **Note:**
>
> In the preceding example, only multi-valued **user attributes**, such as `mail` and `uid` (with an ordering matching rule) are sorted.

18.6 Handling Directory Data

The directory server provides a full set of LDAPv2- and LDAPv3-compliant client tools to manage directory entries. You can add, update, or remove entries by using the `ldapmodify` and `ldapdelete` utilities. The LDAP command-line utilities require LDAP Data Interchange Format (LDIF)-formatted input, entered through the command line or read from a file.

Before you make modifications to directory data, ensure that you understand the following concepts:

- The privilege and access control mechanisms.

  For information about setting privileges, Controlling Access To Data.

- The structure of your directory server.
- The schema of your directory server.

This section contains the following topics:

18.6.1 Adding Directory Entries

You can add one or more entries to a directory server by using the `ldapmodify` command. `ldapmodify` opens a connection to the directory server, binds to it, and performs the modification to the database (in this case, an "add") as specified by the command-line options.

`ldapmodify` enables you to add entries in one of two ways:

- **Using the --defaultAdd option**. Use the `--defaultAdd` option to add new entries to the directory when data is entered on the command line. Press Ctrl-D (UNIX, Linux) or Ctrl-Z (Windows) when finished, or use an input file with your changes.
- **Using LDIF update statements**. LDIF update statements define how `ldapmodify` changes the directory entry. LDIF update statements contain the DN of the entry to be modified, *changetype* that defines how a specific entry is to be modified (add, delete, modify, modrdn), and a series of attributes and their changed values.

> **Note:**
>
> Any newly added entry must conform to the directory's schema. If you add any entry that does not conform to the schema, the server responds with an Object Class Violation error. You can view the details of the error in the `errors` log.

This section contains the following topics:

18.6.1.1 Creating a Root Entry

The root entry is the topmost entry in the directory and must contain the naming context, or root suffix. You can set up the root entry when you first install the directory server using the graphical user interface (GUI) or the command-line. If you install the directory without any data, create a root entry using the `ldapmodify` command with the `--defaultAdd` option.

1. Create the root entry using `ldapmodify`.

```
$ ldapmodify --hostname localhost --port 1389 --defaultAdd \
  --bindDN "cn=Directory Manager" --bindPassword password
dn: dc=example,dc=com
objectclass: domain
objectclass: top
```

```
dc: example
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.

Processing ADD request for dc=example,dc=com
ADD operation successful for DN dc=example,dc=com
```

> **Note:**
>
> The `--bindDN` and `--bindPassword` options specify the bind DN and password, respectively, of the user with permissions to add new entries. You can provide the clear-text version of the password. The server encrypts this value and store only the encrypted one. Be sure to limit read permissions to protect clear passwords that appear in LDIF files. To avoid this security issue, use SSL or startTLS.

2. Verify the change by using the `ldapsearch` command.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope base --bindDN "cn=Directory Manager" --bindPassword password \
  "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

18.6.1.2 Adding an Entry Using the `--defaultAdd` Option With `ldapmodify`

To add an entry using the `--defaultAdd` option, run the ldapmodify command as follows:

1. Create your directory entry in LDIF format.

   Before you add an entry, ensure that the suffix to which you want to add the entry exists in your database (for example, `ou=People,dc=example,dc=com`).

   For this example, create an input file called `new.ldif` with the following contents:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
cn: Marcia Garza
sn: Garza
givenName: Marcia
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
ou: Accounting
ou: People
l: Santa Clara
uid: mgarza
mail: mgarza@example.com
roomnumber: 5484
userpassword: donuts
```

2. Add the entry using `ldapmodify` with the `--defaultAdd` option.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --defaultAdd --filename /tmp/new.ldif
```

18.6.1.3 Adding Entries Using an LDIF Update Statement With `ldapmodify`

To add entries using an LDIF update statement with ldapmodify command, follow these steps:

1. Create the entry in LDIF format with the `changetype:add` element.

   Ensure that there are no trailing spaces after `add`. If a space exists after `add`, the server base-64 encodes the value to represent the space, which can cause problems.

   For this example, create an input LDIF file named `new.ldif`.

   ```
   dn: uid=Marcia Garza,ou=People,dc=example,dc=com
   changetype: add
   cn: Marcia Garza
   sn: Garza
   givenName: Marcia
   objectClass: top
   objectClass: person
   objectClass: organizationalPerson
   objectClass: inetOrgPerson
   ou: Accounting
   ou: People
   l: Santa Clara
   uid: mgarza
   mail: mgarza@example.com
   roomnumber: 5484
   userpassword: donuts
   ```

2. Add the entry using `ldapmodify`.

   Do not include the `-a` option as the `changetype` attribute specifies the action.

   ```
   $ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
     --bindPassword password --filename /tmp/new.ldif

   Processing ADD request for uid=Marcia Garza,ou=People,dc=example,dc=com
   ADD operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
   ```

18.6.2 Adding Attributes

To add attributes to an entry, use the `changetype:modify` statement, as shown in examples in the following sections. You can combine multiple commands within a file by separating each command with a dash ("-").

The LDIF `changetype:add` statement adds an entry to the directory.

This section describes how to manage an entry, and contains the following topics:

- Adding an Attribute to an Entry
- Adding an ACI Attribute
- Adding an International Attribute

18.6.2.1 Adding an Attribute to an Entry

To add an attribute to an entry, follow these steps:

1. Create the entry in LDIF format with the `changetype:modify` element.

   Use the `modify` change type, because you are modifying an existing entry with the addition of a new attribute. Ensure that there are no trailing spaces after `modify`. After the

`changetype`, specify `add: newAttributeName` and, on the following line, the value of the new attribute.

For this example, create an input LDIF file called `add_attribute.ldif`, as follows:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: +1 408 555 8283
```

> **Note:**
>
> To add multiple attributes, separate the attributes with a dash (-), for example:
>
> ```
> dn: uid=Marcia Garza,ou=People,dc=example,dc=com
> changetype: modify
> add: telephonenumber
> telephonenumber: +1 408 555 8283
> -
> add: building
> building: sc09
> ```

2. Add the attribute by using `ldapmodify`.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --filename /tmp/add_attribute.ldif

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

18.6.2.2 Adding an ACI Attribute

You can use `ldapmodify` to add access control instructions (ACIs) to manage access rights for a user's account. For more information, see Controlling Access To Data and ACI Syntax.

The following example allows a user to modify her own directory attributes.

1. Create the LDIF file containing the ACI.

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///uid=Marcia Garza,ou=People,dc=example,dc=com")
 (targetattr="*")(version 3.0; acl "mgarza rights"; allow (write)
 userdn="ldap:///self";)
```

2. Add the attribute by using `ldapmodify`.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --filename /tmp/add_aci.ldif

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

18.6.2.3 Adding an International Attribute

The directory server represents international locales using a language tag in the form *attribute;language-subtype*. For example, `homePostalAddress;lang-jp:address` specifies the postal address with the locale in Japan (`subtype=jp`).

Use `ldapmodify` to add the attribute.

Affix the language subtype, `lang-cc`, where *cc* is the country code.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password
dn: uid=jarrow,ou=People,dc=example,dc=com
changetype: modify
add: homePostalAddress;lang-jp
homePostalAddress;lang-jp: 1-8-15 Azuchimachi, Chuo-ku
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.
```

> **Note:**
>
> If the attribute value contains non-ASCII characters, they must be UTF-8 encoded.

18.6.3 Modifying Directory Entries

Use the LDIF update statement `changetype:modify` to make changes to existing directory data.

The following procedures provide examples of modifying directory entries, and contains the sections:

- Modifying an Attribute Value
- Modifying an Attribute With Before and After Snapshots
- Deleting an Attribute
- Changing an RDN
- Moving an Entry

For more information, see ldapmodify

18.6.3.1 Modifying an Attribute Value

Use `ldapmodify` to change the entry, using the `changetype:modify` and `replace` elements.

Ensure that there are no trailing spaces after `modify`.

This example modifies a user's existing telephone number.

```
$ ldapmodify -h localhost -p 1389 D "cn=Directory Manager" -j pwd-file \
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 8288

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

To modify multiple attributes, separate the attributes with a dash (-), for example:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 6465
```

```
-
add: facsimiletelephonenumber
facsimiletelephonenumber: +1 408 222 4444
-
replace: l
l: Sunnyvale
```

18.6.3.2 Modifying an Attribute With Before and After Snapshots

The `ldapmodify` command provides the options, `--preReadAttribute` and `--postReadAttribute`, that return the modified attribute value with a *before* and *after* snapshot, respectively.

Use `ldapmodify` with the `--preReadAttribute` and `--postReadAttribute` options.

This example modifies a user's existing telephone number.

```
$ ldapmodify -h localhost -p 1389 D "cn=Directory Manager" -j pwd-file \
  --preReadAttributes telephoneNumber --postReadAttributes telephoneNumber
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 8288

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com

Target entry before the operation:
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 4283

Target entry after the operation:
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 8288
```

18.6.3.3 Deleting an Attribute

This example deletes the location (l) attribute from an entry.

Use the `ldapmodify` to delete the attribute.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
delete: l
(Press CTRL-D for Unix, Linux) (Press CTRL-Z for Windows), then press ENTER.

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

Type control-D (UNIX, Linux) or control-Z (Windows) to complete the input.

18.6.3.4 Changing an RDN

The distinguished name (DN) of an entry uniquely identifies and describes that entry. A distinguished name consists of the name of the entry itself as well as the names, in order from bottom to top, of the objects above it in the directory.

The relative distinguished name (RDN) is the leftmost element in an entry DN. For example, the RDN for `uid=Marcia Garza,ou=People,dc=example,dc=com` is `uid=Marcia Garza`. To change an RDN, use the `changetype:moddn` LDIF update statement.

You can specify if the old RDN should be retained in the directory by using the `deleteoldrdn` attribute. A `deleteoldrdn` value of `0` indicates that the existing RDN should be retained in the directory. A value of `1` indicates that the existing RDN should be replaced by the new RDN value.

1. Use the `ldapmodify` command to rename the entry.

   In this example, an employee `Marcia Garza` wants to change to her married name, `Marcia Peters`.

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: uid=Marcia Garza,ou=Marketing,dc=example,dc=com
   changetype: moddn
   newrdn: uid=Marcia Peters
   deleteoldrdn: 1
   Processing MODIFY DN request for uid=Marcia Garza,ou=People,dc=example,dc=com
   MODIFY DN operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
   ```

2. Change any other attributes as necessary.

   In this example, certain attributes might still list the user's previous name.

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: uid=Marcia Peters,ou=People,dc=example,dc=com
   changetype: modify
   replace: sn
   sn: Peters
   -
   replace: cn
   cn: Marcia Peters
   -
   replace: uid
   uid: mpeters
   uid: Marcia Peters
   -
   replace: mail
   mail: mpeters@example.com
   (Press Ctrl-D on Unix, Linux)
   (Press Ctrl-Z on Windows), then press ENTER.

   Processing MODIFY request for uid=Marcia Peters,ou=People,dc=example,dc=com
   MODIFY operation successful for DN uid=Marcia Peters,ou=People,dc=example,dc=com
   ```

18.6.3.5 Moving an Entry

If you are moving an entry from one parent to another, extend the access control instruction (ACI) rights on the parent entries. On the current parent entry of the entry to be moved, ensure that the ACI allows the export operations by using the syntax `allow(export...)`. On the future parent entry of the entry to be moved, ensure that the ACI allows the import operations by using the syntax `allow(import...)`.

In this example, move `uid=sgarza` from the `ou=Contractors,dc=example,dc=com` suffix to the `ou=People,dc=example,dc=com` subtree.

1. Use `ldapmodify` with the `moddn` changetype to move the entry.

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: uid=sgarza,ou=Contractors,dc=example,dc=com
   ```

```
changetype: moddn
newrdn: uid=sgarza
deleteoldrdn: 0
newsuperior: ou=People,dc=example,dc=com
--filename move_entry.ldif
Processing MODIFY DN request for uid=sgarza,ou=Contractors,dc=example,dc=com
MODIFY DN operation successful for DN uid=sgarza,ou=Contractors,dc=example,dc=com
```

2. Change any other attribute values, as required.

   The following example provides before and after snapshot changes for the `ou` attribute.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --preReadAttributes ou --postReadAttributes ou
dn: uid=sgarza,ou=People,dc=example,dc=com
changetype: modify
replace: ou
ou: People
ou: Product Testing
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.

Processing MODIFY request for uid=sgarza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=sgarza,ou=People,dc=example,dc=com

Target entry before the operation:
dn: uid=sgarza,ou=People,dc=example,dc=com
ou: Contractors
ou: Product Testing

Target entry after the operation:
dn: uid=sgarza,ou=People,dc=example,dc=com
ou: People
ou: Product Testing
```

18.6.4 Deleting Directory Entries

You can use `ldapmodify` and `ldapdelete` to remove entries from the directory. The `ldapmodify` command removes entries and attributes by using the LDIF update statements `changetype:delete` and `changetype:modify` with the `delete` attribute, respectively. The `ldapdelete` tool removes only entries.

> **Note:**
>
> You cannot delete an entry that has children entries. If you want to delete an entry that has children, first delete all the children entries below the targeted entry, then delete the entry.

The following topics describe how to delete directory entries:

- Deleting an Entry Using `ldapmodify`
- Deleting an Entry Using `ldapdelete`
- Deleting Multiple Entries Using a DN File

For more information, see ldapdelete.

18.6.4.1 Deleting an Entry Using `ldapmodify`

Use the `ldapmodify` command with the `changetype:delete` statement.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: delete
(Press CTRL-D for Unix)
(Press CTRL-Z for Windows), then press ENTER.

Processing DELETE request for uid=Marcia Garza,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
The number of entries deleted was 1
```

18.6.4.2 Deleting an Entry Using `ldapdelete`

Use the `ldapdelete` command and specify the entry that you want to delete.

```
$ ldapdelete -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
"uid=mgarza,ou=People,dc=example,dc=com"

Processing DELETE request for uid=Marcia Garza,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

18.6.4.3 Deleting Multiple Entries Using a DN File

To delete multiple entries using a DN file, follow these steps:

1. Create a file that contains a list of DNs to be deleted.

   In this example, the file is named `delete.ldif`. The file must list each DN on a separate line, for example:

   ```
   uid=mgarza,ou=People,dc=example,dc=com
   uid=wsmith,ou=People,dc=example,dc=com
   uid=jarrow,ou=People,dc=example,dc=com
   uid=mbean,ou=People,dc=example,dc=com
   ```

2. Delete the entries by passing the file as an argument to the `ldapdelete` command.

   ```
   $ ldapdelete -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --continueOnError --filename delete.ldif

   Processing DELETE request for uid=mgarza,ou=People,dc=example,dc=com
   DELETE operation successful for DN uid=mgarza,ou=People,dc=example,dc=com
   Processing DELETE request for uid=wsmith,ou=People,dc=example,dc=com
   DELETE operation successful for DN uid=wsmith,ou=People,dc=example,dc=com
   Processing DELETE request for uid=jarrow,ou=People,dc=example,dc=com
   DELETE operation successful for DN uid=jarrow,ou=People,dc=example,dc=com
   Processing DELETE request for uid=mbean,ou=People,dc=example,dc=com
   DELETE operation successful for DN uid=mbean,ou=People,dc=example,dc=com
   ```

> **Note:**
>
> The `--continueOnError` option specifies that if an error occurs, the command continues to the next search item.

18.7 Indexing Directory Data

Indexes are configured per server and index configuration is not replicated. You can use `dsconfig` to create local database indexes and Virtual List View (VLV) indexes.

A local database index is used to find entries that match search criteria. A VLV index is used to process searches efficiently with VLV controls.

Unindexed searches are denied by default, unless the user has the `unindexed-search` privilege. For more information, see Changing a Root User's Privileges.

You can determine whether a search is indexed in two ways:

- Try to perform the search anonymously. (The server rejects unindexed anonymous searches by default.)
- Use the `debugsearchindex` operational attribute. This attribute provides the indexes used in the search, the number of candidate entries from each index, and the final indexed status. Include the `debugsearchindex` attribute in your `ldapsearch` command, as follows:

```
$ ldapsearch -h localhost -p 1389 -b "dc=example,dc=com" "(objectClass=*)"
debugsearchindex
```

The following sections describe how to index attributes using the `dsconfig` command-line tool:

- Configuring Indexes on the Local DB Back End
- Configuring VLV Indexes

18.7.1 Configuring Indexes on the Local DB Back End

Understand about the supported index types on the local DB back end and create new local DB index. There are examples included in this section to create and add indexes.

This section contains the following topics:

- Supported Index Types on the Local DB Back End
- Creating a New Local DB Index
- Examples on Creating and Adding Indexes

18.7.1.1 Supported Index Types on the Local DB Back End

The Local DB back end supports the following index types:

- `approximate` — Improves the efficiency of searches using approximate search filters.
- `equality` - Improves the efficiency of searches using equality search filters.
- `ordering` - Improves the efficiency of searches using "greater than or equal to" or "less than or equal to" search filters. In the future, this index type might also be used for server-side sorting.
- `presence` - Improves the efficiency of searches using presence search filters.
- `substring` - Improves the efficiency of searches using substring search filters.

The directory server supports indexing for only a subset of extensible matching operations, including indexes based on collation matching rules and the relative time and partial date and time matching rules. For more information, see Searching Internationalized Entries, and

Understanding Relative Time Matching Rules, and Understanding Partial Date Or Time Matching Rules.

When you create a new local DB back end with `dsconfig`, the following default indexes are created automatically:

- `aci` (presence index)
- `ds-sync-hist` (ordering index)
- `entryuuid` (equality index)
- `objectclass` (equality index)

18.7.1.2 Creating a New Local DB Index

To create a new local DB index, perform the following steps:

> **Note:**
>
> After you have created a new index, you must rebuild the indexes using the `rebuild-index` utility. The directory server cannot use the new index until the indexes have been rebuilt. For more information, see rebuild-index.

1. Create the new index.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-local-db-index \
  --element-name backend --index-name attribute \
  --set index-type:index-type
```

2. Check that the index was created by listing the local DB indexes for that back end.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-local-db-indexes \
  --element-name backend
```

3. Configure any specific index properties.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-local-db-index-prop \
  --element-name backend --index-name attribute \
  --set property:value
```

4. List the index properties to verify your change.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-local-db-index-prop \
  --element-name backend --index-name attribute
```

5. Rebuild the index.

   a. Either stop the server, reb evenuild the index, then restart the server.

```
$ stop-ds
$ rebuild-index --baseDN baseDN --index attribute
$ start-ds
```

   b. Or, rebuild the index online by running the rebuild-index command as a task.

```
$ rebuild-index -h localhost -p 4444 -D "cn=Directory manager" -j pwd-file \
 -X -n --baseDN dc=example,dc=com --index aci
```

```
Rebuild Index task 20110201162742312 scheduled to start immediately
...
Rebuild Index task 20110201162742312 has been successfully completed
```

> **Note:**
>
> Even for an online re-index operation, the back end is unavailable during the re-index. In a replicated topology, the overall service remains available through the referral on update feature. For more information, see Understanding Referrals in a Replicated Topology.

18.7.1.3 Examples on Creating and Adding Indexes

The following examples show how to create a new Equality Index and how to add a Substring Index:

- Creating a New Equality Index
- Adding a Substring Index

18.7.1.3.1 Creating a New Equality Index

The example in this section creates a new equality index for the `employeeNumber` attribute, verifies the index properties, and sets the index entry limit to 5000.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-local-db-index \
  --element-name userRoot --index-name employeeNumber \
  --set index-type:equality

$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-local-db-indexes \
  --element-name userRoot
Local DB Index : Type    : index-type
---------------:---------:-----------
...
employeeNumber : generic : equality
...
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-local-db-index-prop \
  --element-name userRoot --index-name employeeNumber
Property                      : Value(s)
------------------------------:---------------
attribute                     : employeenumber
index-entry-limit             : 4000
index-extensible-matching-rule : -
index-type                    : equality
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-local-db-index-prop \
  --element-name userRoot --index-name employeeNumber --set index-entry-limit:5000
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-local-db-index-prop \
  --element-name userRoot --index-name employeeNumber
Property                      : Value(s)
------------------------------:---------------
attribute                     : employeenumber
index-entry-limit             : 5000
```

```
index-extensible-matching-rule : -
index-type                     : equality
$ rebuild-index -h localhost -p 4444 -D "cn=Directory manager" -j pwd-file -X \
  --baseDN dc=example,dc=com --index employeeNumber
```

18.7.1.3.2 Adding a Substring Index

The example in this section adds a substring index to the index created in the earlier example.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-local-db-index-prop \
  --'
p userRoot --index-name employeeNumber \
  --add index-type:substring
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-local-db-index-prop \
  --element-name userRoot --index-name employeeNumber
Property                       : Value(s)
-------------------------------:---------------
attribute                      : employeenumber
index-entry-limit              : 5000
index-extensible-matching-rule : -
index-type                     : equality, substring
$ rebuild-index -h localhost -p 4444 -D "cn=Directory manager" -j pwd-file -X \
  --baseDN dc=example,dc=com --index employeeNumbe
```

18.7.2 Configuring VLV Indexes

Understand about VLV index configuration and learn to create new VLV index with the help of examples.

The following topics describe about configuring VLV Indexes:

- About VLV Indexes Configuration
- Creating a New VLV Index
- Example of Creating New VLV Index

18.7.2.1 About VLV Indexes Configuration

A VLV index applies to a particular search on a given base entry and its subtree. The sort order, scope of the index, base DN, and filter must be defined when you create the index.

After you have created a new VLV index, you must rebuild the indexes using the `rebuild-index` command, appending `vlv.` in front of the index name. The directory server cannot use the new index until the indexes have been rebuilt. For more information, see rebuild-index.

> **Note:**
>
> Access to the VLV request control is allowed only to authenticated users, by default. If you want to allow unauthenticated users to use the VLV control in search requests, you must change the corresponding global ACIs. For more information, see Allowing Anonymous Access to the Virtual List View Control.

18.7.2.2 Creating a New VLV Index

To create a new VLV Index, follow these steps:

1. Use `dsconfig` to create a new VLV index as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  create-local-db-vlv-index \
  --element-name backend --index-name name --set sort-order:attributes \
  --set scope:scope --set base-dn:baseDN --set filter:filter
```

   where:

   - `index-name` specifies a unique index name, which cannot be altered after the VLV index is created.
   - `sort-order` specifies the names of the attributes by which the entries are sorted and their order of precedence, from highest to lowest.
   - `scope` specifies the LDAP scope of the query being indexed and can be one of `base-object`, `single-level`, `subordinate-subtree`, or `whole-subtree`.
   - `base-dn` specifies the base DN used in the search query being indexed.
   - `filter` specifies the LDAP filter used in the query being indexed and can be any valid LDAP filter.

2. Check that the index was created by listing the existing VLV indexes.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  list-local-db-vlv-indexes \
  --element-name backend
```

3. Display the index properties to verify your change.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-local-db-vlv-index-prop \
  --element-name backend --index-name name
```

4. Rebuild the index.

   a. Either stop the server, rebuild the index, then restart the server.

```
$ stop-ds
$ rebuild-index --baseDN baseDN --index vlv.name
$ start-ds
```

   b. Or, rebuild the index online by running the rebuild-index command as a task.

```
$ rebuild-index -h localhost -p 4444 -D "cn=Directory manager" -j pwd-file -X \
  --baseDN baseDN --index vlv.name
```

18.7.2.3 Example of Creating New VLV Index

The following example creates a new VLV index to sort entries first by surname and then by common name for queries `sn=*`. The example then rebuilds the index online.

```
$ dsconfig -D "cn=directory manager" -j pwd-file -n create-local-db-vlv-index \
  --element-name userRoot --index-name myVLVIndex --set sort-order:"sn cn" \
  --set scope:base-object --set base-dn:dc=example,dc=com --set filter:sn=*
$ rebuild-index -h localhost -p 4444 -D "cn=Directory manager" -j pwd-file -X \
  -b "dc=example,dc=com" --index vlv.myVLVIndex
```

18.8 Reducing Stored Data Size

The directory server provides two mechanisms for reducing the size of stored data. These mechanisms are Compact encoding and Entry compression.

The following topics enables you to understand about reducing stored data size using one of the above mentioned mechanisms and about saving database space using tokens for attribute values:

18.8.1 About Stored Data Size Reduction

You can reduce the size of the stored data by enabling compact encoding and entry compression.

The directory server provides two mechanisms for reducing the size of stored data:

- **Compact encoding**. When compact encoding is enabled, the back end uses a compact form when encoding entries by compressing the attribute descriptions and object class sets. This property applies only to the entries themselves and does not impact the index data. Compact encoding is enabled by default but can be disabled if required. If your deployment requires user-supplied capitalization in object class and attribute type names, you might want to disable compact encoding because user-supplied capitalization is not preserved in compacted entries. The compaction does, however, provide a performance gain and is therefore beneficial in deployments where user-supplied capitalization can be sacrificed for performance, or is not required.
- **Entry compression**. Entry compression uses a deflator to compress the data before it is stored. When entry compression is enabled, the back end attempts to compress entries before storing them in the database. This property also applies only to the entries themselves and does not impact the index data. The effectiveness of entry compression is based on the type of data contained in the entry.

You can enable one or both of these mechanisms to reduce the size of the stored data. Because enabling these mechanisms affects future writes only, the database might contain a mixture of compressed and uncompressed records. Either type of record can be read regardless of the compression settings.

18.8.2 Enabling or Disabling Compact Encoding

Compact encoding is configured by setting the `compact-encoding` property of a Local Backend workflow element. Changes to this setting will only take effect for writes that occur after the change is made. Existing data is not changed retroactively.

Disable compact encoding on the "userRoot" workflow element.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-workflow-element-prop --element-name="userRoot" --set compact-encoding:false
```

18.8.3 Enabling or Disabling Entry Compression

Entry compression is configured by setting the `entries-compressed` property of a Local Backend workflow element. Changes to this setting will only take effect for writes that occur after the change is made. Existing data is not changed retroactively.

Enable entry compression on the "userRoot" back end.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-workflow-element-prop --element-name="userRoot" \
  --set entries-compressed:true
```

18.8.4 Saving Database Space Using Tokens for Attribute Values

Oracle Unified Directory server can compact attributes that have a small number of values and are repeated a large number of times in many entries. The server references these attributes and their values using tokens. The server stores the tokens and their values once in a separate table and then stores only the tokens in the database entries. This optimization saves space when attribute values would otherwise be repeated across many database entries. It can also improve cache efficiency when the entire database does not otherwise fit in memory.

For example, consider a telco with a `mobile-phone` user attribute that stores the names of mobile phones and models used by all customers (users). Every user entry in the telco's database has a `mobile-phone` attribute value (if not several values). The set of values the `mobile-phone` can acquire is limited, and the most popular `mobile-phone` values are repeated thousands (or even millions) of times across the users in the database.

However, using this optimization, Oracle Unified Directory server uses tokens to store the `mobile-phone` attribute values. The server stores the tokens with the attribute values only once in a separate table and then stores the tokens in the database entries.

To configure a list of attributes that should be compacted using tokens, set the multivalued `ds-cfg-compact-attribute-values-using-tokens` property in the DB Local Backend workflow element. For example:

```
$ dsconfig set-workflow-element-prop \
           --element-name userRoot \
           --set compact-attribute-values-using-tokens:mobile-phone \
           --hostname host --port admim-port \
           --trustStorePath INSTALL_PATH/asinst_1/OUD/config/admin-truststore \
           --bindDN "cn=Directory Manager" \
           --bindPasswordFile ****** --no-prompt
```

After you set this property, changes take effect only for writes that occur after the change. Existing data is not changed retroactively.

To compact existing data for an attribute that preceded the configuration using the `ds-cfg-compact-attribute-values-using-tokens` property, you can export and then re-import that data (although you should consider the cost of this operation against the performance you will gain by using a token for the attribute).

After you compact attributes using the `ds-cfg-compact-attribute-values-using-tokens` property, the `dbtest` command displays the total number of tokens that have been created in the database to store the values. For example:

```
$ dbtest list-database-containers -n userRoot

Database Name               Database Type  JE Database Name          Entry Count
```

```
--------------------------------------------------------------------------------
Schema maps
compressed_object_classes                                                    8
compressed_attributes                                                       32
compressed_values                                                           14
...
```

The `compressed_values` line shows that in this deployment, 14 values have been encoded as tokens and can be reused for the encoding of a large number of values. You can use `dbtest` to check that this feature has been configured correctly and that the number of tokens does not become too large.

18.8.5 Retrieving Multi-Valued Attributes in the Order of Creation

You can store and retrieve multi-valued attributes in the order in which they are created.

You can store multiple values for an attribute. However, because of the compact encoding feature, whenever multi-valued attributes are queried, there is no guarantee that multiple values of the attribute would be returned in the order in which they are created. While retrieving multiple attribute values, if you want the values to be returned in the same order in which they are created, then you should disable compact encoding. Refer to Enabling or Disabling Compact Encoding on how to disable compact encoding.

18.9 Configuring Selective Attribute Caching

You can use selective attribute caching to reduce memory requirements for larger deployments and improve performance when working with large entries.

The following topics describe how you can use selective attribute caching:

- Understanding Selective Attribute Caching
- Example of Using Selective Attribute Caching
- Configuring Attribute-Level Caching
- Monitoring Cold Attributes Usage

18.9.1 Understanding Selective Attribute Caching

Oracle Unified Directory performs I/O and database caching on the entire LDAP entry for read or write operations. However, most read and write operations only target specific attributes and rarely access other attributes being stored in the database. For larger deployments and large entries, this behavior can impact memory and performance.

Selective attribute caching enables you to better manage these operations by differentiating the attributes in an LDAP entry, based on how often they are accessed:

- **Regular attributes**: Attributes that are frequently accessed. For example, office phone numbers, user IDs, or email addresses.
- **Cold attributes**: Attributes that are rarely accessed. For example, pager numbers, home phone numbers, or binary data such as jpeg photos.

  You can configure cold attributes that work only on operational demand or that fit certain use cases.

> **Note:**
>
> You must have a very good understanding of your applications to effectively designate cold attributes.

Be aware that regular and cold attributes may differ for various LDAP application workloads. For example, if your deployment rarely accesses employee pager numbers or home phone numbers, you could configure those attributes as cold attributes. However, another customer's deployment might frequently access employee pager numbers and phone numbers, so in their case it would be inappropriate to configure those attributes as cold.

> **Note:**
>
> Although there are no restrictions on which attributes you can configure as cold, designating as cold any attributes used by various core server features (such as the following) could negate the benefit of selective attribute caching and cause unexpected behavior in some core server features.
>
> - Groups (definitions could be based on an entry attribute)
> - Virtual attributes (they could depend on other entry attributes)
> - Password policy attributes (they read and change policy attributes)
> - User Account Notification
> - Assertion Control
> - Persistent Search
>
> Designating these core server attributes as cold is not supported.

After specifying cold attributes, the server then splits the LDAP entry store into two databases, `id2entry` and `id2entry-cold`. The server stores regular attributes in the regular `id2entry` database and cold attributes in the `id2entry-cold` database. Using two databases reduces I/O for operations on the partial entry data and reduces the memory footprint for the java heap, database cache, and file system (FS) cache.

18.9.2 Example of Using Selective Attribute Caching

Learn more about using selective attribute caching.

Consider a scenario where your LDAP entry store contains the following attributes:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
givenName: Aaccf
sn: Amar
cn: Aaccf Amar
employeeNumber: 0
uid: user.0
mail: user.0@example.com
userPassword: password
```

```
telephoneNumber: +1 024 705 1954
homePhone: +1 021 391 6930
mobile: +1 195 481 7233
initials: AFA
street: 77569 Lake Street
l: Elmira
st: ND
postalCode: 31858
postalAddress: Aaccf Amar$77569 Lake Street$Elmira, ND 31858
description: This is the description for Aaccf Amar.
pager: +1 575 339 1600
```

You might decide to configure and store the attributes as follows:

id2entry Database	id2entry-cold Database
dn: uid=user.0,ou=People,dc=example,dc=com objectClass: top objectClass: person objectClass: organizationalperson objectClass: inetorgperson givenName: Aaccf sn: Amar cn: Aaccf Amar employeeNumber: 0 uid: user.0 mail: user.0@example.com userPassword: password telephoneNumber: +1 024 705 1954 homePhone: +1 021 391 6930 mobile: +1 195 481 7233	initials: AFA street: 77569 Lake Street l: Elmira st: ND postalCode: 31858 postalAddress: Aaccf Amar$77569 Lake Street$Elmira, ND 31858 description: This is the description for Aaccf Amar. pager: +1 575 339 1600

Oracle Unified Directory caches attributes from the `id2entry-cold` database only if they are accessed and caching priority is given to regular attributes over cold attributes. Consequently, operations that do not target cold attributes are more likely to get into the cache.

Also, if write operations are not targeting any cold attributes, then those operations do not have to rewrite the id2entry-cold database, which makes them faster particularly when larger attributes are declared as cold attributes.

> **Note:**
>
> Selective attribute caching *might not* improve search performance for databases that completely fit in the DB cache. However, selective attribute caching allows you to increase cache hits for the most commonly used attributes in the cases where the database does not completely fit in memory.

18.9.3 Configuring Attribute-Level Caching

The attribute-level caching is related to the DB cache, hence the configuration stays in the back-end configuration entry.

To configure cold attributes in the DB Local Backend Workflow Element, use the multi-valued **`ds-cfg-cold-attribute`** property to specify the names of cold attributes for your database.

> **Note:**
>
> You can specify any attributes as cold attributes, but you should avoid specifying any attributes that your server relies on for processing your operations; such as ACI, password policy, and other previously mentioned core server features.

For example,

```
dsconfig set-workflow-element-prop \
          --element-name userRoot \
          --add cold-attribute:description \
          --add cold-attribute:initials \
          --add cold-attribute:l \
          --add cold-attribute:pager \
          --add cold-attribute:postalAddress \
          --add cold-attribute:postalCode \
          --add cold-attribute:st \
          --add cold-attribute:street \
```

After executing this `dsconfig` command, the `userRoot` workflow element properties (as displayed by `dsconfig`) will look like the following:

```
Property                     : Value(s)
-----------------------------:-------------------------------
1) base-dn                   : "dc=example,dc=com"
2) cold-attribute            : description, initials, l, pager, postalAddress,
                               postalCode,st, street
...
```

After executing the `dsconfig` command, the `userRoot` workflow element entry under `cn=config` will look like the following:

```
dn                           : cn=userRoot,cn=Workflow Elements,cn=config
objectClass                  : ds-cfg-local-backend-workflow-element
objectClass                  : ds-cfg-workflow-element
objectClass                  : ds-cfg-db-local-backend-workflow-element
objectClass                  : top
...
ds-cfg-cold-attribute        : description
ds-cfg-cold-attribute        : pager
ds-cfg-cold-attribute        : postalCode
ds-cfg-cold-attribute        : postalAddress
ds-cfg-cold-attribute        : st
ds-cfg-cold-attribute        : l
ds-cfg-cold-attribute        : street
ds-cfg-cold-attribute        : initials
```

> **Note:**
>
> When you define cold attributes, the server does not move any existing data into the cold database. The server starts storing cold attributes in the cold database only when you add a new entry or modify an existing entry.
>
> Ideally, do a fresh import of your data after defining cold attributes.

Restrictions

When using cold attributes, the following restrictions apply:

- As previously mentioned, designating as cold any attributes that are used by the core server features (such as access control instructions (ACIs) or virtual ACIs, password policy attributes, groups, etc.) could negate the benefit of selective attribute caching and could lead to unexpected behavior of some core server features. Designating these attributes as cold is not supported.
- Oracle Unified Directory cannot cache entries with cold attributes in the entry cache. Attribute-level caching is useful when the deployment is constrained by memory and the entry cache should not be used because it is very costly in memory.

18.9.4 Monitoring Cold Attributes Usage

Oracle Unified Directory provides a Cold Attributes Usage monitor that keeps track of each time the server accesses a cold attribute. This monitor is disabled by default to prevent a performance hit, but you can enable it for diagnostic purposes by using the `ds-cfg-monitor-cold-attributes` backend configuration property.

After configuring cold attributes, you can enable the monitor and run the server using specific application workloads. The monitor records which cold attributes were accessed by the server and how many times they were accessed. You can use this data to refine your cold attributes configuration. For example, if monitoring shows that a cold attribute has been accessed a large number of times, then you may want to reconsider reconfiguring it as a regular attribute.

In the following sample output, you can see that the `description` attribute was accessed nine times and that all of the other cold attributes were accessed three times.

For Example, consider the following sample monitoring output:

```
dn: cn=dc_example_dc_com Cold Attributes Usage,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
cn: dc_example_dc_com Cold Attributes Usage
l: 3
st: 3
initials: 3
postalCode: 3
pager: 3
description: 9
postalAddress: 3
street: 3
```

18.10 Ensuring Attribute Value Uniqueness

A directory's structure requires that distinguished names be unique to identify the object and its place in the directory information tree. The directory server provides a *Unique Attribute* plug-in, which ensures that the value of an attribute is unique when the attribute is added, modified, or moved within the directory.

This section describes the following topics:

- Overview of the Unique Attribute Plug-In
- Configuring the Unique Attribute Plug-In Using `dsconfig`
- Ensuring Unique Attribute Values in a Replication Environment

18.10.1 Overview of the Unique Attribute Plug-In

The unique attribute plug-in is disabled by default. You can enable the plug-in by using the `dsconfig` command and can define the suffix and attributes that it should check. When it is enabled, the plug-in identifies whether an LDAP add, modify, or modify DN operation causes two entries to have the same attribute value before the database is updated by the operation. If the server recognizes a conflict, the operation is terminated and an `LDAP_CONSTRAINT_VIOLATION` error is returned to the client.

When you enable attribute uniqueness on an existing directory, the server does not check for uniqueness among existing entries. After the plug-in is enabled, uniqueness is enforced when an entry is added, modified, or moved.

You can configure the unique attribute plug-in to enforce uniqueness in one or more subtrees in the directory or among entries of a specific object class. You can define several instances of the unique attribute plug-in if you want to enforce the uniqueness of other attributes. Typically, you define one plug-in instance for each attribute whose value must be unique. You can also have several plug-in instances for the same attribute to enforce "separate" uniqueness in several sets of entries.

The unique attribute plug-in is disabled by default, so that multi-master replication configuration is not affected. When the plug-in is enabled, it checks that the `uid` attribute is unique prior to any add, modify, or modify DN operations for stand-alone systems and checks for uniqueness after synchronization in replicated environments.

Like other plug-ins, the unique attribute plug-in is configured by using the `dsconfig` command. For more information, see Configuring Plug-Ins Using `dsconfig`. The easiest way to configure plug-ins is to use `dsconfig` in interactive mode. Interactive mode functions like a wizard and walks you through the plug-in configuration. Because the interactive mode is self-explanatory, the examples in this section do not demonstrate interactive mode, but provide the equivalent complete `dsconfig` commands.

18.10.2 Configuring the Unique Attribute Plug-In Using `dsconfig`

Learn to configure the attribute value uniqueness using `dsconfig` command.

This section contains the following topics:

- Ensuring Uniqueness of the `uid` Attribute Value
- Ensuring Uniqueness of Any Other Attribute Value

> **Note:**
>
> See Using a Password File With Server Commands to learn how to create a password file, for example `pwd-file`, used in the preceding procedures.

18.10.2.1 Ensuring Uniqueness of the `uid` Attribute Value

The unique attribute plug-in checks the `uid` attribute by default. The following task enables the unique attribute plug-in, and sets the base DN under which attribute value uniqueness for the `uid` attribute should be checked.

1. Display the plug-ins that are currently defined in the server.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  list-plugins
```

Depending on your installation, the output will be similar to the following.

```
Plugin                          : Type                            : enabled
--------------------------------:---------------------------------:--------
7-Bit Clean                     : seven-bit-clean                 : false
Change Number Control           : change-number-control           : true
Entry UUID                      : entry-uuid                      : true
LastMod                         : last-mod                        : true
LDAP Attribute Description List : ldap-attribute-description-list : true
Password Policy Import          : password-policy-import          : true
Profiler                        : profiler                        : true
Referential Integrity           : referential-integrity           : false
UID Unique Attribute            : unique-attribute                : false
```

2. Display the properties that are configured for the unique attribute plug-in

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-plugin-prop \
  --plugin-name "UID Unique Attribute" \
Property : Value(s)
---------:---------
base-dn  : -
enabled  : false
type     : uid
```

3. Enable the unique attribute plug-in.

```
$ dsconfig --advanced -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop \
  --plugin-name "UID Unique Attribute" --set enabled:true
```

> **Note:**
>
> Ensure that you run the `dsconfig` command with `--advanced` subcommand. This subcommand modifies the display output to show the advanced plug-ins like `postaddoperation`, `postmodifyoperation`, and `postmodifydnoperation` that can be selected. The default values are pre-operation plug-ins like `preaddoperation`, `premodifyoperation`, and `postmodifyoperation`. You must select a matching pre-operation plug-in with a post-operation plug-in.

4. Set the base DN under which uniqueness is checked.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop \
  --plugin-name "UID Unique Attribute" --set base-dn:ou=People,dc=example,dc=com
```

18.10.2.2 Ensuring Uniqueness of Any Other Attribute Value

The unique attribute plug-in checks the `uid` attribute by default. If you want to ensure uniqueness for a different attribute, create a new instance of the unique attribute plug-in and set its `type` property.

This example creates a new instance of the unique attribute plug-in and ensures uniqueness of the `mail` attribute.

1. Create and enable a new instance of the unique attribute plug-in.

Set the `type` property to the name of the attribute that should be unique (in this case, `mail`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  create-plugin \
  --type unique-attribute --plugin-name "MAIL unique attribute"
  --set enabled:true --set type:mail
```

2. Enable the new unique attribute plug-in.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set enabled:true
```

3. Set the base DN under which uniqueness is checked.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set base-dn:ou=People,dc=example,dc=com
```

4. Specify the attribute whose value must be unique.

   This example specifies the `mail` attribute.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set type:mail
```

To ensure that the values of more than one attribute are unique, create and enable multiple instances of the unique attribute plug-in.

18.10.3 Ensuring Unique Attribute Values in a Replication Environment

The Unique Attribute plug-in does not check attribute uniqueness when an update is performed as part of a replication operation.

To ensure attribute value uniqueness in a replication environment, enable the unique attribute plug-in for the same attribute in the same subtree on all servers in the topology. It is recommended that you direct all the updates to a single server which then replicates.

18.11 Configuring Virtual Attributes

Virtual attributes are attributes whose values do not exist in persistent storage but are dynamically generated.

You can configure virtual attributes by using the `dsconfig` command or using the OUDSM graphical user interface, as described in the following sections:

- Supported Virtual Attributes
- Configuring Virtual Attributes Using `dsconfig`
- Configuring Virtual Attributes Using OUDSM

18.11.1 Supported Virtual Attributes

Understand about the supported virtual attributes from the following tabular column.

Oracle Unified Directory supports the following virtual attribute types:

Table 18-3 Supported Virtual Attributes

Virtual Attribute Name	Description
`collective attribute subentries`	Generates a virtual attribute that specifies all collective attribute subentries that affect the entry.
`entryDN`	Generates the entryDN operational attribute in directory entries, which contains a normalized form of the entry's DN.
`entryUUID`	Ensures that all entries contained in private back ends have values for the entryUUID operational attribute.
`governingStructureRule`	Specifies the DIT structure rule with the schema definitions in effect for the entry.
`hasSubordinates`	Indicates whether the entry has any subordinate entries.
`isMemberOf`	Contains the DNs of the groups in which the user is a member.
`member`	Generates a member or `uniqueMember` attribute whose values are the DNs of the members of a specified virtual static group.
`nsuniqueid`	Generates a unique identifier that is assigned to each entry in the directory server to resolve naming conflicts while migrating legacy applications using Oracle Directory Server Enterprise Edition as an LDAP database to Oracle Unified Directory.
`numSubordinates`	Specifies the number of immediate child entries that exist below the entry.
`orclguid`	Creates an orclguid virtual attribute.
`Password Expiration Time`	Indicates the exact time after which the user's password expires. You can issue a `SEARCH` operation to read that specific user entry *and* explicitly request the server to return the `passwordExpirationTime` attribute for that entry. If the `passwordExpirationTime` attribute is enabled, then the value is computed and returned in the search result through that attribute.
`Password Policy Subentry`	Points to the Password Policy subentry in effect for the entry.
`Proximity`	Specifies location based proximity in meters.
`structuralObjectClass`	Specifies the structural object class with the schema definitions in effect for the entry.
`subschemaSubentry`	Specifies the location of the subschemaSubentry with the schema definitions in effect for the entry.
`User-defined`	Creates virtual attributes with user-defined values in entries that match the criteria defined in the plug-in's configuration.

18.11.2 Configuring Virtual Attributes Using `dsconfig`

The easiest way to configure virtual attributes using `dsconfig` is in interactive mode. Interactive mode functions like a wizard and walks you through the virtual attribute configuration. Because the interactive mode is self-explanatory, the examples in this section do not demonstrate interactive mode, but provide the equivalent complete `dsconfig` commands.

The following topics describe how to configure and manage virtual attributes using the `dsconfig` command:

- Listing the Existing Virtual Attributes Using `dsconfig`
- Creating a New Virtual Attribute Using `dsconfig`

- Enabling or Disabling a Virtual Attribute Using `dsconfig`
- Viewing the Configuration of a Virtual Attribute Using `dsconfig`
- Changing the Configuration of a Virtual Attribute Using `dsconfig`

For more information about using `dsconfig`, see Managing the Server Configuration Using `dsconfig`.

18.11.2.1 Listing the Existing Virtual Attributes Using `dsconfig`

The directory server provides several virtual attribute rules by default.

To view a list of all configured virtual attribute rules, run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file
-n list-virtual-attributes
```

The following example shows a sample output of the above command which is a list of all configured virtual attribute rules.

```
Virtual Attribute               : Type                             : enabled : attribute-type
--------------------------------:----------------------------------:---------:----------------
Collective Attribute Subentries : collective-attribute-subentries : true    :
collectiveattributesubentries
entryDN                         : entry-dn                         : true    : entrydn
entryUUID                       : entry-uuid                       : true    : entryuuid
governingStructureRule          : governing-structure-rule         : true    : governingstructurerule
hasSubordinates                 : has-subordinates                 : true    : hassubordinates
isMemberOf                      : is-member-of                     : true    : ismemberof
nsuniqueid                      : nsuniqueid                       : true    : nsuniqueid
numSubordinates                 : num-subordinates                 : true    : numsubordinates
orclguid                        : orclguid                         : true    : orclguid
Password Expiration Time        : password-expiration-time         : true    : passwordexpirationtime
Password Policy Subentry        : password-policy-subentry         : true    : pwdpolicysubentry
Proximity                       : proximity                        : true    : proximity
structuralObjectClass           : structural-object-class          : true    : structuralobjectclass
subschemaSubentry               : subschema-subentry               : true    : subschemasubentry
Virtual Static member           : member                           : true    : member
Virtual Static uniqueMember     : member                           : true    : uniquemember
```

The following information (from left to right) is included in the above listed sample output of the command:

- `Virtual Attribute`. Displays the name of the virtual attribute, which is usually descriptive of what it does.
- `Type`. Displays the type of virtual attribute. You can define more than one virtual attribute of a specific type.
- `enabled`. Indicates whether the virtual attribute is enabled or disabled. Disabled virtual attributes remain in the server configuration, but their values are never generated.
- `attribute-type`. Displays the type of attribute for which the virtual values are generated.

18.11.2.2 Creating a New Virtual Attribute Using `dsconfig`

To create new virtual attributes, use the `create-virtual-attribute` subcommand.

For example, you could run the following `dsconfig` command to create and enable a virtual attribute rule that adds a virtual fax number of `+61 2 45607890` to any user entry with a location of Sydney (unless they already have a fax number in their entry):

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  create-virtual-attribute \
  --type user-defined --name "Sydney Fax Number" \
  --set attribute-type:facsimiletelephonenumber --set enabled:true \
  --set value:+61245607890 --set filter:"(&(objectClass=person)(l=Sydney))"
```

18.11.2.3 Enabling or Disabling a Virtual Attribute Using `dsconfig`

To enable a virtual attribute, set the `enabled` property to **`true`**. To disable a virtual attribute, set the `enabled` property to **`false`**.

For example, run the following command to disable the virtual attribute created in the previous example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-virtual-attribute-prop --name="Sydney Fax Number" --set enabled:false
```

18.11.2.4 Viewing the Configuration of a Virtual Attribute Using `dsconfig`

To display the configuration of a virtual attribute, use the `get-*-prop` subcommand.

For example, run the following command to view a list of properties for the virtual attribute created in Creating Virtual Attributes Using OUDSM:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-virtual-attribute-prop --name="Sydney Fax Number"
Property          : Value(s)
------------------:---------------------------------
attribute-type    : facsimiletelephonenumber
base-dn           : -
conflict-behavior : real-overrides-virtual
enabled           : false
filter            : (&(objectClass=person)(l=Sydney))
group-dn          : -
value             : +61245607890
```

18.11.2.5 Changing the Configuration of a Virtual Attribute Using `dsconfig`

To change the configuration of a virtual attribute, use the `set-*-prop` subcommand.

For example, you could use this command to change the behavior of a virtual attribute when a conflict occurs. By default, the value of a real attribute overwrites the virtual attribute value. Running the following command, merges the real attribute value and the virtual attribute value.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-virtual-attribute-prop --name="Sydney Fax Number" \
  --set conflict-behavior:merge-real-and-virtual
```

18.11.3 Configuring Virtual Attributes Using OUDSM

You can manage virtual attributes using OUDSM.

The following topics describe how to display and create virtual attributes by using the Configuration tab in OUDSM:

- Listing Existing Virtual Attributes Using OUDSM

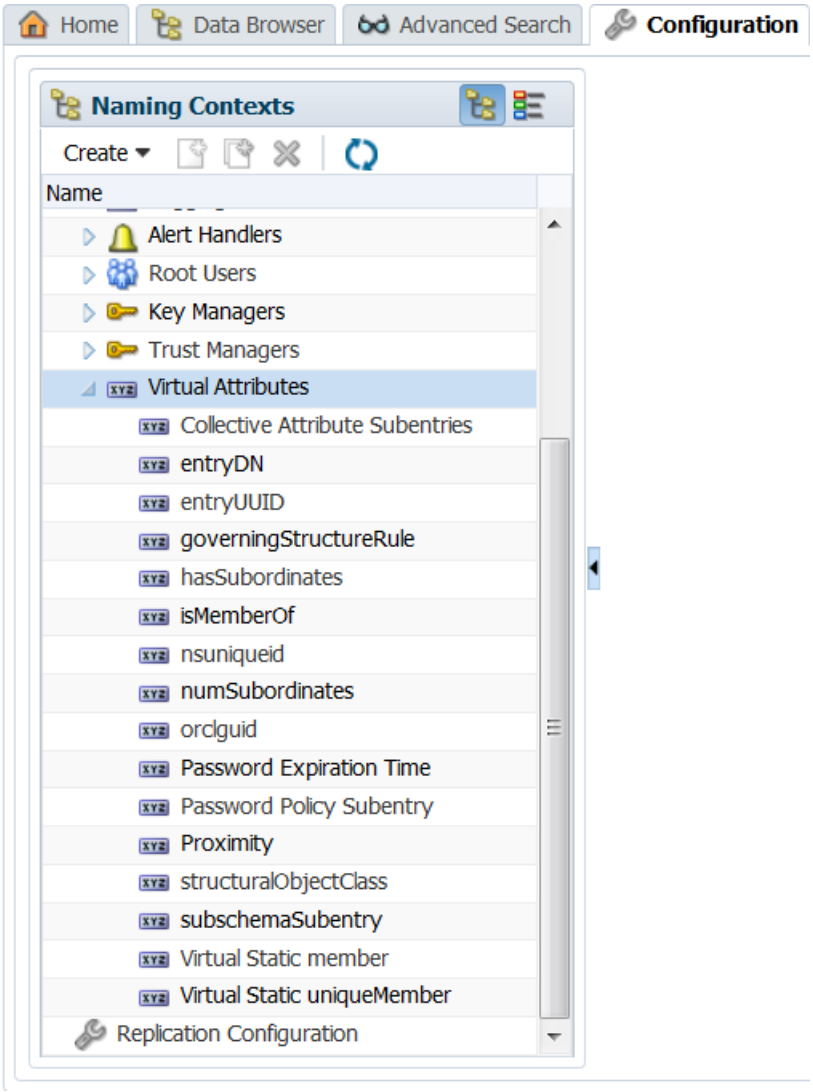
- Creating Virtual Attributes Using OUDSM
- Viewing the Configuration of a Virtual Attribute Using OUDSM
- Changing the Configuration of a Virtual Attribute Using OUDSM
- Enabling or Disabling a Virtual Attribute Using OUDSM

18.11.3.1 Listing Existing Virtual Attributes Using OUDSM

To view a list of existing virtual attributes with OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** node.
4. Expand the attributes in the **Virtual Attributes** node to display all the existing virtual attributes.

Figure 18-1 Virtual Attributes



5. Click the virtual attribute name to view detailed information about that attribute in the right-hand pane.

18.11.3.2 Creating Virtual Attributes Using OUDSM

To create a virtual attribute:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. From the Create menu, select **Virtual Attributes.**
4. In the **Name** field, type the name of the virtual attribute.
5. The **Enabled** box is checked by default indicating that the virtual attribute will be enabled.

   To disable this virtual attribute later, return to this page and clear the box.
6. From the **Virtual Attribute Type** list, select the type of virtual attribute that you want to create.
7. Use the **Attribute Type** Select menu to specify an attribute type for the attribute whose values are to be dynamically assigned by the virtual attribute.
8. Click the **Add** icon to enter the Base DN for the branches containing entries that are eligible to use this virtual attribute.

   Do one of the following to enter the Base DN:

   - In the Base DN field, type the desired Base DN.
   - Click **Select** to use the Tree view or the Search view to select entries.
9. Click the **Add** icon to specify the DNs of the group whose members are eligible to use this virtual attribute.

   Do one of the following to specify the DNs of the group:

   - In the Group DN field, type the desired Group DN.
   - Click **Select** to use the Tree view or the Search view to select entries.
10. Click the **Add** icon to specify the search filters to apply against these entries to determine if a virtual attribute must be generated for those entries.
11. **For User Defined virtual attributes only**, configure the following additional properties:

    - **Conflict Behavior:** Specifies the behavior that the server has to exhibit for entries that already contain one or more real values for the associated attribute. It has the following values:

      Merge real and virtual: Indicates that the virtual attribute provider is to preserve any real values contained in the entry and merge them with the set of generated virtual values so that both the real and virtual values are used.

      Real overrides virtual: Indicates that any real values contained in the entry are preserved and used, and virtual values are not generated.

      Virtual overrides real: Indicates that the virtual attribute provider suppresses any real values contained in the entry, and generates virtual values and uses them.
    - **Value:** Specifies the values to be included in the virtual attribute.
12. **For Member virtual attributes only**, configure the following additional properties:

    - **Conflict Behavior:** It is similar to the User Defined Virtual Attributes.

- **Allow Retrieving Membership:** Indicates whether to handle requests that demands all values for the virtual attribute. The default value is `false`.

13. Click **Create.**

18.11.3.3 Viewing the Configuration of a Virtual Attribute Using OUDSM

To view the configuration settings of a virtual attribute:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Click the **Core Configuration** icon.
4. Expand the **Virtual Attributes** list and select the virtual attribute for which you want to view the configuration settings.

   The configuration settings are displayed on the right

18.11.3.4 Changing the Configuration of a Virtual Attribute Using OUDSM

To change the configuration settings of a virtual attribute:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Click the **Core Configuration** icon.
4. Expand the **Virtual Attributes** list and select the virtual attribute that you want to edit.
5. When the attribute's configuration page displays on the right, modify the settings as needed.

   If necessary, refer back to the configuration instructions described in Creating Virtual Attributes Using OUDSM.
6. Click **Apply**.

18.11.3.5 Enabling or Disabling a Virtual Attribute Using OUDSM

You can enable or disable a virtual attribute by opening the attribute's configuration page (as described in Changing the Configuration of a Virtual Attribute Using OUDSM) and using the **Enabled** box:

- To enable the virtual attribute, check the box.
- To disable the virtual attribute, clear the box.

18.12 Using LDAP Subentries

LDAP subentries are special entries that hold operational data for the server, and have the `ldapSubEntry` object class. They are similar to operational attributes in that they are not returned to clients unless explicitly requested by including a Subentries Control request control.

This section includes the following topics:

- About LDAP SubentriesAbout LDAP Subentries
- Relative Subtrees

18.12.1 About LDAP Subentries

LDAP subentries can be used to specify a range of entries. This functionality is used in the definition of collective attributes and can also be useful in other areas like access control.

For more information, see Using Collective Attributes and Defining a Password Policy as an LDAP Subentry.

A subtree specification uses the following parameters to define the set of entries:

- Base

  This is the relative name of the root of the subtree relative to the administrative point. So, if the administrative point is `ou=system` and the base is `ou=users`, the subtree begins at `ou=users,ou=system`. The base can be any length of name components, including "". In this case, the subtree begins at the administrative point, `ou=system` in the previous example.

- Chop

  The `chopBefore` and `chopAfter` parameters are names relative to the base of the subtree, that specify whether an entry and its descendants should be excluded from the collection.The `minimum` parameter describes the minimum number of name components between the base and the target entry required to include entries within the selection. The `maximum` parameter describes the maximum length between the base and the target allowed before entries are excluded from the collection.

- Specification filter

  The specification filter refines the subtree that has been defined by the previous parameters so that it is not a contiguous set of entries but rather a set of collected entries based on the `objectClass` characteristics of the entries.

  For example, you can define a subtree to cover a region of an administrative area but include only `inetOrgPersons` within this region.

The Oracle Unified Directory implementation of LDAP subentries is based on RFC 3672 (`http://www.ietf.org/rfc/rfc3672.txt`), with one extension - relative subtrees, described in the following section.

18.12.2 Relative Subtrees

Relative subtrees function like standard LDAP subtrees, except that the specification filter is not a set of refinements but an LDAP search filter.

For relative subtree specification ensure that you use the relativeBase keyword to specify the root of the subtree. Do not use the base keyword to specify the root of the subtree.

For example, the following subtree definition targets all users under the base DN `ou=People`, whose location is Paris:

```
subtreeSpecification: {relativeBase "ou=people", specificationFilter "(l=Paris)" }
```

18.13 Using Collective Attributes

Collective attributes are attributes whose values are shared across a collection of entries. Collective attributes provide similar functionality to the Oracle Directory Server Enterprise Edition Class of Service feature.

Oracle Unified Directory collective attributes are like virtual attributes but are defined and stored with the user data as LDAP subentries. As part of the user data, collective attributes can be replicated to other servers in the topology.

The following sections describe the collective attribute implementation in Oracle Unified Directory and explains how to configure collective attributes:

- Extensions to the Collective Attributes Standard
- Configuring Collective Attributes
- Overview of Inherited Collective Attributes

18.13.1 Extensions to the Collective Attributes Standard

The Oracle Unified Directory implementation of collective attributes is based on RFC 3671 and RFC 3672 with a few specific extensions. These extensions make Oracle Unified Directory collective attributes more transparent for LDAP client applications.

See RFC 3671 (`http://www.ietf.org/rfc/rfc3671.txt`) and RFC 3672 (`http://www.ietf.org/rfc/rfc3672.txt`). Oracle Unified Directory collective attributes are described in the following sections:

- About Collective Attributes Naming
- Example of Using Collective Attributes Naming and Conflict Resolution
- Excluding Collective Attributes From Specific Entries

18.13.1.1 About Collective Attributes Naming

According to RFC 3671 (`http://www.ietf.org/rfc/rfc3671.txt`), collective attributes *must* have the `COLLECTIVE` attribute type, be derived from regular user attributes defined in the schema, and have the `c-` prefix. For example, `c-l` is a collective attribute for the standard `l` attribute, and affected user entries have `c-l` added to them as needed.

This specification can cause problems for many client applications, which are typically not aware of collective attributes and might need to be modified or extended to handle collective attributes. Oracle Unified Directory therefore removes this restriction and supports the definition of any regular attribute defined in the schema as a collective attribute. This extension is facilitated by adding the required attribute to the related collective attribute subentry and marking the attribute with the collective option.

18.13.1.2 Example of Using Collective Attributes Naming and Conflict Resolution

Collective attributes can be named in various ways. Consequently, a conflict resolution mechanism is provided for affected user entries already containing related real attributes. Oracle Unified Directory provides the same conflict resolution options for collective attributes as it does for virtual attributes: `real-overrides-virtual`, `virtual-overrides-real`, and `merge-real-and-virtual`.

The default conflict resolution rule is `real-overrides-virtual`. If an entry already has the same attribute type defined, the explicitly defined attribute takes precedence over the collective attribute. This behavior can be changed for each collective attribute subentry (to `virtual-overrides-real` or `merge-real-and-virtual`) by using the `collectiveConflictBehavior` attribute.

The following example dynamically adds the `l` collective attribute with a value of `Paris` to each applicable user entry under `ou=people`. The value of the collective attribute overrides any value for `l` that is specific to the entry:

```
dn: cn=People Locale,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
cn: People Locale
l;collective: Savoie
subtreeSpecification: {base "ou=people", minimum 1}
collectiveConflictBehavior: virtual-overrides-real
```

18.13.1.3 Excluding Collective Attributes From Specific Entries

In some instances, it might be necessary to avoid having collective attributes in specific user entries. You can add the `collectiveExclusions` operational attribute to such entries to achieve this behavior. To exclude specific collective attributes, list the attribute names as values of the `collectiveExclusions` attribute. To exclude all collective attributes, set the value of `collectiveExclusions` to `excludeAllCollectiveAttributes`.

The following example excludes the `preferredLanguage` attribute from being applied to the entry for `user.0`:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectclasses and other user attributes
collectiveExclusions: preferredLanguage
```

The following example excludes the `c-l` attribute from being applied to the entry for `user.1`:

```
dn: uid=user.1,ou=People,dc=example,dc=com
objectclasses and other user attributes
collectiveExclusions: c-l
```

The following example excludes both the `preferredLanguage` and `c-l` attributes from being applied to the entry for `user.2`:

```
dn: uid=user.2,ou=People,dc=example,dc=com
objectclasses and other user attributes
collectiveExclusions: preferredLanguage
collectiveExclusions: c-l
```

The following example excludes all collective attributes from being applied to the entry for `user.0`:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectclasses and other user attributes
collectiveExclusions: excludeAllCollectiveAttributes
```

18.13.2 Configuring Collective Attributes

Learn about configuring and managing collective attributes.

This section contains the following topics:

- Handling Collective Attributes Configuration
- Creating a New Collective Attribute
- Deleting a Collective Attribute

- Listing the Collective Attributes That Apply to an Entry

18.13.2.1 Handling Collective Attributes Configuration

Collective attributes are defined using LDAP subentries within the directory tree where they are applicable. The following examples use a simple tree with multiple user entries.

```
dn: dc=example,dc=com
    dn: ou=People,dc=example,dc=com
        dn: uid=user.0,ou=People,dc=example,dc=com
        dn: uid=user.1,ou=People,dc=example,dc=com
        dn: uid=user.2,ou=People,dc=example,dc=com
        ...
```

To add a common `preferredLanguage` attribute for all users, create and add a collective attribute subentry similar to the following:

```
dn: cn=People Preferred Language,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
cn: People Preferred Language
preferredLanguage;collective: fr
subtreeSpecification: {base "ou=people", minimum 1}
```

The `preferredLanguage` attribute-value pair is dynamically added to all user entries under `ou=people`, as shown in the following example:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectclasses and other user attributes
preferredLanguage: fr

dn: uid=user.1,ou=People,dc=example,dc=com
objectclasses and other user attributes
preferredLanguage: fr

...
```

The same procedure applies for *collective* attribute types. For example, the `c-l` collective attribute type specifies a locality name for a collection of entries. The following example adds a common `c-l` collective attribute:

```
dn: cn=People Locale,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
cn: People Locale
c-l: Paris
subtreeSpecification: {base "ou=people", minimum 1}
```

The `c-l: Paris` attribute is added to applicable entries, as shown in this example:

```
dn: uid=user.0,ou=People,dc=example,dc=com
objectclasses and other user attributes
c-l: Paris

dn: uid=user.1,ou=People,dc=example,dc=com
objectclasses and other user attributes
c-l: Paris
```

...

You can define multiple collective attributes in the subentry of any collective attribute in the following ways:

- By adding the *collective* attribute types to the subentry
- By adding regular attribute types with the collective option
- By adding a combination of the two

Collective attribute subentries allow for flexible and complex definitions. For information about collective attribute scoping and the `subtreeSpecification` syntax, see RFC 3671 (http://www.ietf.org/rfc/rfc3671.txt) and RFC 3672 (http://www.ietf.org/rfc/rfc3672.txt).

18.13.2.2 Creating a New Collective Attribute

To enable you to create a new collection attribute, follow these steps:

1. Create an LDIF file with the `changetype: add` element that specifies the collective attribute subentry.

   Ensure that there are no trailing spaces after `add`. If a space exists after add, the server base-64 encodes the value to represent the space, which can cause problems.

   This example uses an input LDIF file named `add_collective_attr.ldif`.

   ```
   dn: cn=People Preferred Language,dc=example,dc=com
   changetype: add
   objectClass: top
   objectClass: subentry
   objectClass: collectiveAttributeSubentry
   objectClass: extensibleObject
   cn: People Preferred Language
   preferredLanguage;collective: fr
   subtreeSpecification: {base "ou=people", minimum 1}
   ```

2. Use the `ldapmodify` command to add the collective attribute, as shown in the following example.

   ```
   $ ldapmodify -p 1389 -h localhost -D "cn=Directory Manager" -j pwd-file \
   -f /usr/local/add_collective_attr.ldif
   Processing ADD request for cn=People Preferred Language,dc=example,dc=com
   ADD operation successful for DN cn=People Preferred Language,dc=example,dc=com
   ```

18.13.2.3 Deleting a Collective Attribute

You can delete a collective attribute by using either the `ldapdelete` command or the `ldapmodify` command.

Use the `ldapmodify` command with the `changetype: delete` element, as shown in the following example.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
  dn: cn=People Preferred Language,dc=example,dc=com
  changetype: delete
  deleting entry cn=People Preferred Language,dc=example,dc=com
```

18.13.2.4 Listing the Collective Attributes That Apply to an Entry

To list the collective attribute subentries that apply to a specific user entry, request the `collectiveAttributeSubentries` operational attribute for that entry.

Use the `ldapsearch` command to list the collective attribute subentries that apply to the `user.0` entry:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "uid=user.0,ou=People,dc=example,dc=com" \
  "objectclass=*" "collectiveAttributeSubentries"

  version: 1
  dn: uid=user.0,ou=People,dc=example,dc=com
  collectiveAttributeSubentries: cn=People Preferred Language,dc=example,dc=com
```

18.13.3 Overview of Inherited Collective Attributes

Inherited attributes enable a common set of attributes to be shared by nature of their inheritance. Inherited collective attributes provide flexible scoping mechanisms using the standard subentry subtree specification, and support any attribute type for RDN definition and construction.

This section contains the following topics about inherited collective attributes:

- About Inherited Collective Attributes
- Specifying Inherited Collective Attributes

18.13.3.1 About Inherited Collective Attributes

The main difference between collective attributes and inherited collective attributes is the source of attribute values:

- A collective attribute always derives its value from its definition entry.
- An inherited collective attribute can inherit the collective attribute values from other entities, either directly or indirectly.

The inherited collective attributes functionality is built upon and extends collective attributes. Inherited attributes are defined as a specific type of collective attribute subentry (`inheritedCollectiveAttributeSubentry`). This type is further divided into the following two distinct subtypes:

- `inheritedFromDNCollectiveAttributeSubentry`
- `inheritedFromRDNCollectiveAttributeSubentry`

Each subtype has its own set of configuration attributes. The subtypes cannot be mixed in a single definition, so an inherited attribute definition can be of only one subtype.

Entries that are under the scope of an inherited collective attribute entry can potentially point to multiple "template" entries and can therefore inherit values for the inheritAttribute from multiple entries. In this case, the first value that is processed takes precedence.

As with other virtual attributes, no schema checking is performed on inherited attributes. Inheritance can, therefore, result in entries that violate the schema. However, since these attributes are all virtual, this kind of schema violation can be ignored as it does not have an impact on server function.

Inherited collective attributes provide similar functionality to the Oracle Directory Server Enterprise Edition Class of Service (Classic CoS). For example, suppose you have the following user entry:

```
uid=psmith,ou=people,dc=example,dc=com
departmentNumber: 123
...
```

the following department entry:

```
cn=123,ou=departments,dc=example,dc=com
telephoneNumber: 4486152643
...
```

and the following inherited attribute definition:

```
dn: cn=classicCOS,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: inheritedCollectiveAttributeSubentry
objectClass: inheritedFromRDNCollectiveAttributeSubentry
cn: classicCOS
subtreeSpecification: {base "ou=people"}
inheritFromBaseRDN: ou=departments
inheritFromRDNAttribute: departmentNumber
inheritFromRDNType: cn
inheritAttribute: telephoneNumber
```

The inherited collective attribute sub-entry would apply to user entries under `ou=people,dc=example,dc=com`. The `telephoneNumber` attribute would be added to each of these entries. The value of the `telephoneNumber` attribute would be inherited from the entry whose DN is constructed with the following logic:

inheritFromRDNType=inheritFromRDNAttribute,*inheritFromBaseRDN*,"*inherited collective attribute sub-entry rootDN*"

or `cn=123,ou=departments,dc=example,dc=com`

The affected user entries would therefore be of the form:

```
uid=psmith,ou=people,dc=example,dc=com
departmentNumber: 123
...
telephoneNumber: 4486152643
```

18.13.3.2 Specifying Inherited Collective Attributes

Like regular collective attributes, inherited collective attributes are defined using LDAP subentries within the directory tree where they are applicable.

The following examples use a simple tree with multiple user entries.

```
dn: dc=example,dc=com
     dn: ou=People,dc=example,dc=com
         dn: uid=hpollock,ou=People,dc=example,dc=com
         dn: uid=cventer,ou=People,dc=example,dc=com
         dn: uid=sdonnelly,ou=People,dc=example,dc=com
         ...
```

To add an inherited `postalAddress` attribute for all users, create and add an inherited collective attribute subentry similar to the following:

```
dn: cn=indirectCOS,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: inheritedCollectiveAttributeSubentry
objectClass: inheritedFromDNCollectiveAttributeSubentry
cn: indirectCOS
subtreeSpecification: {base "ou=people"}
inheritFromDNAttribute: manager
inheritAttribute: postalAddress
```

This subentry specifies that the user entry inherits its `postalAddress` value from the entry referenced by the `manager` attribute in the user's entry.

The manager's entry contains the real value for the `postalAddress` attribute:

```
dn: uid=dsmith,ou=People,dc=example,dc=com
... objectclasses and other user attributes
postalAddress: 650 Granger Parkway, Redwood Shores, CA 94065
```

Each user entry references the manager entry, and inherits its `postalAddress` from that entry:

```
dn: uid=hpollock,ou=People,dc=example,dc=com
... objectclasses and other user attributes
manager: uid=dsmith,ou=People,dc=example,dc=com
postalAddress: 650 Granger Parkway, Redwood Shores, CA 94065

dn: uid=cventer,ou=People,dc=example,dc=com
... objectclasses and other user attributes
manager: uid=dsmith,ou=People,dc=example,dc=com
postalAddress: 650 Granger Parkway, Redwood Shores, CA 94065

dn: uid=sdonnelly,ou=People,dc=example,dc=com
... objectclasses and other user attributes
manager: uid=dsmith,ou=People,dc=example,dc=com
postalAddress: 650 Granger Parkway, Redwood Shores, CA 94065
```

18.14 Configuring Referrals

A *referral* is a pointer to a remote suffix or entry that is returned to a client instead of a result.

The following topics describe how to configure referrals:

- Overview of Configuring Referrals
- Understanding Referrals in a Replicated Topology
- Configuring the Referral List Manually
- Managing Smart Referrals
- Understanding LDAP URLs

18.14.1 Overview of Configuring Referrals

When a server cannot handle a client's request, it sends a list of referrals to the client, which point the client to other servers in the topology. The client then performs the operation again on one of the remote servers in the referral list. Understand about the Referrals from this topic.

The server returns a list of referrals in the following cases:

- Writability is disabled or set to `internal-only` on the server or on the Local Backend workflow element. For more information, see writability mode.

This kind of referral is called *referral on update*.

- The Local Backend workflow element has been placed in maintenance mode.

  You can place a Local Backend workflow element in maintenance mode if you want to prevent the server from responding to client requests temporarily.

  To place a back end in maintenance mode, set the `maintenance` property of the Local Backend workflow element to `true`.

- The back end is unavailable for some reason, for example a data import or re-index is in process.
- The client request specifically targets a *smart referral*. For more information, see Managing Smart Referrals.

A referral URL is an LDAP URL that includes the host name, port number, and optionally a DN on the local host or on another server. For more information, see Understanding LDAP URLs.

The server returns the result code `REFERRAL (10)` along with a list of referral URLs, if available. If no referral URLs are available, the server returns the result code `UNAVAILABLE (52)`.

The list of referral URLs can be created in two ways:

- For replicated servers, use the replication service to propagate the list. For more information, see Understanding Referrals in a Replicated Topology.
- Create the list manually by setting the `ds-cfg-referrals-url` property of the DB Local Backend workflow element. For more information, see Configuring the Referral List Manually.

18.14.2 Understanding Referrals in a Replicated Topology

The replication service generates a list of referral URLs to which requests can be redirected. This list corresponds to the LDAP/LDAPS connection handlers configured on each local server. To publish a value other than the LDAP/LDAPS connection handler, you can define your own referral URLs as values of the `referrals-url` property of the replication domain on the local server.

When a client request targets a replicated server that is unavailable, the server sends the list of referral URLs to which the request can be redirected.

The list of referral URLs is organized according to the protocol that was used for the request. For example, if an operation is done over LDAPS, the first URLs that are provided are those that use the same secure protocol (LDAPS).

In addition, the list is organized by groupID. The URLs that represent a server in the same replication group are presented first. The list of URLs is limited to 16 URLs for each protocol type (LDAP/LDAPS) and excludes any untrusted servers.

For security considerations, referrals that are propagated by the replication service are not returned on untrusted servers. Untrusted servers should not divulge information about the rest of the topology. If a client request targets an untrusted server, the list of referral URLs will only include the servers that are managed by the administrator on the local back end. In addition, the referral URLs that are provided by the replication service exclude any untrusted servers in the topology.

If the `publish-referrals` configuration property of a replication domain is set to false, that server will not be included in the list of referrals that is generated by the replication service.

18.14.3 Configuring the Referral List Manually

To override the list of referral URLs that is presented by the replication service, or to set up referrals outside of a replicated topology, set the `referrals-url` property of the DB Local Backend workflow element. The `referrals-url` property takes one or more LDAP URLs as values.

The following example specifies that any client requests targeting the `dc=example,dc=com` suffix should be referred to the server running on the host `host1.example.com` and listening on port 2389.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n
\
 set-workflow-element-prop --element-name userRoot \
 --set referrals-url:ldap://host1.example.com:2389/dc=example,dc=com
```

To specify multiple LDAP URLs, use the `--add` suboption multiple times. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n
\
 set-workflow-element-prop --element-name userRoot \
 --add referrals-url:ldap://host1.example.com:2389/dc=example,dc=com
 --add referrals-url:ldap://host2.example.com:1389/dc=example,dc=com
```

18.14.4 Managing Smart Referrals

A smart referral is a special type of entry that references content on another server or in another suffix. Smart referral entries contain the **referral** object class with one or more instances of the `ref` attribute. Each `ref` attribute contains an LDAP URL that is used in the referral.

This section contains the following topics:

- Configuring a Smart Referral
- Modifying a Smart Referral
- Deleting a Smart Referral

18.14.4.1 Configuring a Smart Referral

To configure a smart referral, add a new entry that contains a `referral` object class and a `ref` attribute. The `ref` attribute must contain an LDAP URL.

This example creates a referral on server B for a user entry that exists on server A.

1. Locate the user entry on server A by running the following search command:

   ```
   $ ldapsearch -h serverA -p 1389 -b dc=example,dc=com "uid=user.199" cn
   dn: uid=user.199,ou=People,dc=example,dc=com
   cn: Alfred Altay
   ```

2. Add a referral entry to the directory on server B.

   ```
   $ ldapmodify -h serverB -p 2389 -D "cn=directory manager" -j pwd-file
   dn: uid=aaltay,ou=People,dc=example,dc=com
   changetype: add
   objectclass: top
   objectclass: extensibleObject
   objectclass: referral
   ```

```
uid: aaltay
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)

Processing ADD request for uid=aaltay,ou=People,dc=example,dc=com
ADD operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

3. As a user with sufficient access rights, search for the user entry on server B.

```
$ ldapsearch -h serverB -p 2389 -D "cn=directory manager" -j pwd-file \
  -b dc=example,dc=com "uid=aaltay"
SearchReference(referralURLs={ldap://localhost:1389/dc=example,dc=com??sub?})
```

18.14.4.2 Modifying a Smart Referral

To view or modify a smart referral, use the `ldapsearch` or `ldapmodify` commands with the `manageDsaIT` control. This control informs the server that you intend to manage the referral object as a regular entry and prevents the server from sending a referral result for requests that read or update referral objects.

1. Use the `ldapsearch` command to view the referral.

```
$ ldapsearch -h serverB -p 2389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com --control managedsait "(uid=aaltay)" ref
dn: uid=aamar,ou=People,dc=example,dc=com
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)
```

2. Use the `ldapmodify` command to modify the referral.

   This example changes the server to which the referral points and the base DN under which the entry is located.

```
$ ldapmodify -h serverB -p 2389 -D "cn=Directory Manager" -j pwd-file \
  --control managedsait
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: modify
replace: ref
ref: ldap://serverC:1389/ou=People,dc=example,dc=com??sub?(uid=user.199)
Processing MODIFY request for uid=aaltay,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

18.14.4.3 Deleting a Smart Referral

To delete a smart referral, use the `ldapdelete` command with the `manageDsaIT` control. This control informs the server that you intend to manage the referral object as a regular entry and prevents the server from sending a referral result for requests that read or update referral objects.

1. Use the `ldapsearch` command to view the referral.

```
$ ldapsearch -h serverB -p 2389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com --control managedsait "(uid=aaltay)" ref
dn: uid=aamar,ou=People,dc=example,dc=com
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)
```

2. Use the `ldapdelete` command to delete the referral.

```
$ ldapdelete -h serverB -p 2389 -D "cn=Directory Manager" -j pwd-file \
  --control managedsait "uid=aaltay,ou=People,dc=example,dc=com"
Processing DELETE request for uid=aaltay,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

18.14.5 Understanding LDAP URLs

RFC 4516 describes the format of an LDAP URL, which is summarized as follows:

```
ldap[s]://hostname:port/base_dn?attributes?scope?filter
```

An LDAP URL includes the following components:

- ldaps[s]

  Indicates whether to connect to the server (`ldap:`), or connect to the server over SSL (`ldaps:`).

- hostname

  Specifies the host name or IP address of the LDAP server.

- port

  Specifies the port number of the LDAP server. If no port is specified, the default LDAP port (389) or LDAPS port (636) is used.

- base_dn

  Specifies the distinguished name (DN) of an entry in the directory. This DN identifies the entry that is the starting point of the search. If no base DN is specified, the search starts at the root of the directory tree.

- attributes

  Returns the specified attributes. Use commas to separate more than one attribute. If no attributes are specified, the search returns all attributes.

- scope

  Specifies the scope of the search:

  – **base.** Search only the base entry specified by *base_dn*.

  – **one.** Search one level below the base entry specified by base_dn

  – **sub.** Search the base entry and all entries below the specified base_dn

  If no scope is specified, the server performs a base search.

- filter

  Specifies the search filter to apply to entries within the specified scope of the search. If no filter is specified, the server uses the default (`objectclass=*`).

  Any spaces must be escaped using a character appropriate to your shell.

> **Note:**
>
> Unless an LDAP client provides authentication, any search request initiated by means of an LDAP URL is anonymous (unauthenticated).

The following is a list of example LDAP URLs:

- The following LDAP URL specifies a search for all entries that have the surname `Jensen` at any level under `dc=example,dc=com`. No port is specified, so the default (`389`) is used. No attributes are specified, so all attributes will be returned.

```
ldap://example.com/dc=example,dc=com??sub?(sn=Jensen)
```

- The following LDAP URL specifies a search for the `cn` and `telephoneNumber` attributes at any level under `dc=example,dc=com`. The server contacts the remote server at port `2389`. Because no search filter is specified, the server uses the default filter `(objectclass=*)`.

  ```
  ldap://example.com:2389/dc=example,dc=com?cn,telephoneNumber?sub
  ```

18.15 Managing Data Using OUDSM

The Data Browser tab of each server instance in OUDSM enables you to perform a basic search on the directory data, and to add, delete, and modify entries.

OUDSM includes an "auto-suggest" facility that enables you to enter a subset of characters in any of the data fields. OUDSM then returns all entries that match that subset of characters. The auto-suggest feature returns only those entries that have already been cached by OUDSM.

The following sections describe how to manage data with OUDSM:

- Viewing Entries
- Viewing the Attributes of an Entry
- Searching for Entries
- Adding an Entry
- Adding an Entry Based on an Existing Entry
- Deleting an Entry
- Deleting an Entry and Its Subtree
- Modifying an Entry's RDN
- Importing Data From an LDIF File
- Exporting Data to an LDIF File

18.15.1 Viewing Entries

You can view the entries and restrict entries to a specific entry set from the **Entry** pane in the **Data Browser** tab.

To view directory entries by using the OUDSM data browser, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Data Browser** tab.
3. Select the appropriate network group from the **Network Group** list.
4. Expand the entries in the **Entry** pane to display all of the entries in the required subtree.

   A maximum of 200 entries is displayed at a time.
5. To restrict the entries to a specific entry set, select the subtree (for example, `ou=People`) and click the **Filter** icon.

   In the **Filter** field, type the required filter (for example, `surname=a*`) and click **OK**.
6. Select the entry that you want to view in the left hand pane.

   The entry details are displayed in the tabs on the right.

See also Viewing the Attributes of an Entry.

18.15.2 Viewing the Attributes of an Entry

You can view the attributes of an entry and learn about the different types of entries from this topic.

To view the attributes of an entry, follow these steps:

1. Display the entry as described in Viewing Entries.
2. Select the entry that you want to view in the left hand pane.

   The entry details are displayed in the tabs on the right.

   Every entry has a corresponding Properties tab, that displays all the possible attributes of the entry (mandatory and optional). In addition, the following types of entries have a customized tab that displays the mandatory attributes of the entry in a layout that is logical for the entry type:

   - `inetorgperson` entries have a corresponding User Page tab.
   - `group` entries have a corresponding Group Page tab.
   - `country` entries have a corresponding Country Page tab.
   - `domain` entries have a corresponding Domain Page tab.
   - `organization` entries have a corresponding Organization Page tab.
   - `organization unit` entries have a corresponding Organization Unit Page tab.

18.15.3 Searching for Entries

The basic search function on the Data Browser tab enables you to search for user or group entries.

To perform a basic search on the directory data, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Data Browser** tab.
3. Select the appropriate network group from the **Network Group** list.
4. Select the **Search** tab on the left hand pane.
5. From the **For** list, select whether you are searching for a user entry or a group entry.
6. Enter any part of the entry name and click the right arrow button. For example, to search for user John Smith, you might enter `Smith`, or `Smi`, or `John`, and so forth.
7. When the entry is displayed in the left pane, double-click the entry to display its details in the right pane.

18.15.4 Adding an Entry

To add or delete entries with Oracle Unified Directory Services Manager, you must have write access to the parent entry and you must know the DN to use for the new entry.

Follow the steps to add an entry:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Data Browser** tab.
3. Select the appropriate network group from the **Network Group** list.
4. Click the **Add Entry** icon and select the kind of entry that you want to add, for example User Entry.
5. Enter the DN of the parent entry. This is the entry beneath which the new entry will appear in the directory tree, for example, `ou=people,dc=example,dc=com`.

   To select an existing entry as the parent entry, click **Select**.

   In the **Entry Picker** window, select **Tree View** to navigate the directory tree and locate the entry, or **Search View** to search for the entry.
6. Enter any additional information for the new entry.
7. When the required details have been entered, click **Create**.

18.15.5 Adding an Entry Based on an Existing Entry

You can add an entry based on the existing entry by using the **Create like entry** option.

To add an entry that is based on an existing entry by using the OUDSM data browser, follow these steps:

1. Display the existing entries as described in Viewing Entries.
2. Select the entry on which you want to base the new entry and click the **Create like entry** icon.

   The details of the existing entry are displayed in the right pane.
3. Provide a new **Common Name** and **User Name** for the entry.
4. Modify any other details of the entry.
5. Click **Create**.

18.15.6 Deleting an Entry

You can delete an entry from the OUDSM data browser by using **Delete** option.

To delete an entry by using the OUDSM data browser, follow these steps:

1. Display the existing entries as described in Viewing Entries.
2. Select the entry that you want to delete and click the **Delete** icon.
3. On the Delete Entry dialog, verify that you are deleting the correct entry and click **OK**.

18.15.7 Deleting an Entry and Its Subtree

You can delete an entry and all entries beneath it in the directory tree by using the **Delete Entry and its Subtree** option.

To delete an entry and its subtree, follow these steps:

1. Display the existing entries as described in Viewing Entries.
2. Select the entry that you want to delete and click the **Delete Entry and its Subtree** icon.

3. On the Delete Subtree dialog, verify that you are deleting the correct entry and click **OK**.

18.15.8 Modifying an Entry's RDN

You can modify the RDN of an entry by using the OUDSM data browser.

Perform the following steps:

1. Display the existing entries as described in Viewing Entries.
2. Select the entry whose RDN you want to modify on which you want to base the new entry and click the **Edit RDN** icon.
3. Provide a new RDN in the **New RDN value** field.
4. Select **Delete Old RDN** if you want the values that formed the old RDN to be deleted from the entry. If you do not select this checkbox, the values that formed the old RDN are retained as non-distinguished attribute values of the entry.
5. Optionally, click the **Refresh subtree entries** icon to verify the RDN change.

18.15.9 Importing Data From an LDIF File

You can import data from an LDIF file using the **Import LDIF** option in the OUDSM **Data Browser**.

To import entries from an LDIF file, follow these steps:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Data Browser** tab.
3. Select the appropriate network group from the **Network Group** list.
4. Click the **Import LDIF** icon.
5. On the **Import Entry(ies)** dialog, click **Choose File**.
6. Locate the LDIF file on your system and click **OK**.
7. On the **LDIF Import Progress** dialog, monitor the progress of the import and click **OK** when the export has completed.
8. The Data Browser tree refreshes to show the new entries.

18.15.10 Exporting Data to an LDIF File

You can export data to an LDIF file by using the **Export Operational Attributes** in the OUDSM data browser

To export entries to an LDIF file, by using the OUDSM data browser, follow these steps:

1. Display the entries as described in Viewing Entries.
2. Navigate to the top level DN of the subtree you want to export and click the **Export LDIF** icon.
3. On the **Export Entry** dialog, select **Export Operational Attributes** if you want the operational attributes to be exported.
4. Click **OK**.
5. **Click here to open the LDIF file**.

The complete LDIF file is displayed in a separate tab of the browser window in which OUDSM is running.

6. Save the LDIF file to a writable location.
7. Click **OK** on the Export Entry dialog to exit the export.

19

Managing Users and Groups

Oracle Unified Directory provides a comprehensive user management model that includes identity mapping, and account status notification by using Oracle Unified Directory Services Manager (OUDSM) interface.
The following topics describe how to configure these elements by using the command-line utilities and by using the Oracle Unified Directory Services Manager (OUDSM) interface:

- Managing User Accounts
- Configuring Root Users
- Defining Groups
- Maintaining Referential Integrity
- Simulating ODSEE Roles in an Oracle Unified Directory Server

For information about user passwords, see Managing Password Policies.

19.1 Managing User Accounts

User accounts are essentially user entries that you create, modify, or remove in your directory. You can manage user accounts and passwords by using command-line utilities.

Before you begin to manage user accounts, ensure that you have the appropriate password policies set up on the directory server. For more information, see Managing Password Policies.

The following topics describe how to manage user accounts and passwords by using the `manage-account` and `ldappasswordmodify` command-line utilities:

- Changing Passwords
- Managing a User's Account Information
- Assigning Resource Limits on a User Account

19.1.1 Changing Passwords

Directory administrators are often asked to create, reset, or remove passwords for other users. Changing passwords can be done by using `ldappasswordmodify` utility.

The `ldappasswordmodify` utility enables you to change or reset a user's password with the LDAP password modify extended operation. You can specify authorization IDs with the `--authzid` option by prefixing `dn:`, `u:`, or by specifying the full DN.

The following topics describe how to manage passwords:

- Changing the Directory Manager's Password
- Resetting and Generating a New Password for a User
- Changing a User's Password

19.1.1.1 Changing the Directory Manager's Password

To change the Directory Manager's password, use the `ldappasswordmodify` command.

Use the `ldappasswordmodify` command, as shown in the following example:

```
$ ldappasswordmodify -h localhost -p 1389 \
  --authzID "dn:cn=Directory Manager" \
  --currentPassword password --newPassword mynewpassword

The LDAP password modify operation was successful.
```

19.1.1.2 Resetting and Generating a New Password for a User

This example assumes that the user does not remember the existing password.

Use the `ldappasswordmodify` command, as shown in the following example:

```
$ ldappasswordmodify -h localhost -p 1389 -D "cn=Directory Manager" \
  -j pwd-file --authzID u:jvedder

The LDAP password modify operation was successful
Generated Password:  evx07npv
```

19.1.1.3 Changing a User's Password

This example assumes that the user remembers the existing password. The new password is passed to the server in a specified file.

Use the `ldappasswordmodify` command, as shown in the following example:

```
$ ldappasswordmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --authzID uid=jvedder,ou=People,dc=example,dc=com \
  --currentPassword password --newPasswordFile pwdFile

The LDAP password modify operation was successful
```

19.1.2 Managing a User's Account Information

The `manage-account` command is used to display information about the user's account and any password policy that is applied to the user.

You can also use `manage-account` command to enable or disable a user's account. The `manage-account` command accesses the server over SSL through the administration port. For more information, see Managing Administration Traffic to the Server.

The following topics describe how to manage a user's account information:

- Viewing a User's Account Information
- Viewing Account Status Information
- Disabling an Account
- Enabling an Account
- Enabling an Account Using `orclIsEnabled`

19.1.2.1 Viewing a User's Account Information

Use **manage-account** command to view a User's account information.

The `manage-account` command returns the DN of the password policy in effect on a user account, as well as the account status, and password and login related information.

1. To display all available information on a user account, use the `manage-account` command with the `get-all` subcommand, as shown in the following example:

```
$ manage-account -D "cn=directory manager" -j pwd-file get-all \
  --targetDN uid=kvaughan,ou=People,dc=example,dc=com
Password Policy DN:  cn=Default Password Policy,cn=Password Policies,cn=config
Account Is Disabled:  false
Account Expiration Time:
Seconds Until Account Expiration:
Password Changed Time:  19700101000000.000Z
Password Expiration Warned Time:
Seconds Until Password Expiration:  432000
Seconds Until Password Expiration Warning:  0
Authentication Failure Times:
Seconds Until Authentication Failure Unlock:
Remaining Authentication Failure Count:
Last Login Time:
Seconds Until Idle Account Lockout:
Password Is Reset:  false
Seconds Until Password Reset Lockout:
Grace Login Use Times:
Remaining Grace Login Count:  4
Password Changed by Required Time:
Seconds Until Required Change Time:
Password History:
```

2. To display just a single property of the account, substitute the `get-all` subcommand with the subcommand corresponding to the property you want to view.

   For example, to view just the password history, run the following command:

```
$ manage-account -D "cn=directory manager" -j pwd-file get-password-history \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
```

   For a complete list of subcommands, run the following command:

```
$ manage-account --help
```

19.1.2.2 Viewing Account Status Information

To assess whether an account is enabled or disabled, use the **manage-account** command.

Use the `manage-account` command with the `get-account-is-disabled` subcommand, as shown in the following example:

```
$ manage-account -D "cn=directory manager" -j pwd-file get-account-is-disabled \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  false
```

19.1.2.3 Disabling an Account

To disable an account, use **manage-account** command.

To disable an account, use the `manage-account` command with the `set-account-is-disabled` subcommand, as shown in the following example:

```
$ manage-account -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-account-is-disabled --operationValue true \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  true
```

19.1.2.4 Enabling an Account

To enable an account, use **manage-account** command.

To enable an account, use the `manage-account` command with the `clear-account-is-disabled` subcommand, as shown in the following example:

```
$ manage-account -D "cn=directory manager" -j pwd-file clear-account-is-disabled \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  false
```

19.1.2.5 Enabling an Account Using `orclIsEnabled`

You can enable an account using the **orclIsEnabled** command.

To enable Oracle Unified Directory using `orclIsEnabled`:

1. Create and enable a new workflow element as follows:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n /
     create-workflow-element --element-name fawe --type fa \
     --set enabled:true --set next-workflow-element:userRoot
   ```

2. Assign the new workflow element to the default workflow, as follows:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n /
     set-workflow-prop --workflow-name userRoot0 --set workflow-element:fawe
   ```

19.1.3 Assigning Resource Limits on a User Account

Assigning resource limits on a user account is done by adding specific operational attributes to the user entry.

The following topics provide conceptual description of the resource limits and describe how to set those limits on a user account:

- About Resource Limits on a User Account
- Setting Resource Limits on a User Account

19.1.3.1 About Resource Limits on a User Account

You can control search operations on the server for each client account by assigning resource limits to the entry. Resource limits are assigned by adding specific operational attributes to the user entry. The directory server then enforces the limits based on the account that the client uses to bind to the directory.

The resource limits that you set on specific user accounts take precedence over the resource limits set in the server-wide configuration. For details of all the configurable resource limit properties, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

The following limits can be set:

- **Look-through limit**. Specifies the maximum number of entries examined for a search operation. Use the `ds-rlim-lookthrough-limit` operational attribute.
- **Size limit**. Specifies the maximum number of entries returned in response to a search operation. use the `ds-rlim-size-limit` operational attribute.
- **Time limit**. Specifies the maximum time spent processing a search operation. Use the `ds-rlim-time-limit` operational attribute.

> **Note:**
>
> The Directory Manager can use unlimited resources by default.

19.1.3.2 Setting Resource Limits on a User Account

You can set resource limits on a user account by modifying an entry in an LDIF file.

To set resource limits on a user account:

1. Modify the entry in an LDIF file, adding the operational attributes, as shown here:

```
dn: uid=kvaughan,ou=people,dc=example,dc=com
changetype: modify
add: ds-rlim-lookthrough-limit
ds-rlim-lookthrough-limit: 1000
-
add: ds-rlim-size-limit
ds-rlim-size-limit: 500
-
add: ds-rlim-time-limit
ds-rlim-time-limit: 300
```

2. Use the `ldapmodify` command to apply the changes, as shown here:

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --filename add_resource.ldif
Processing MODIFY request for uid=kvaughan,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=kvaughan,ou=people,dc=example,dc=com
```

19.2 Configuring Root Users

Configuring root users is done by using the command-line utilities and OUDSM interface.

- About Root Users
- Configuring Root Users Using the Command-Line Utilities
- Configuring Root Users Using OUDSM

19.2.1 About Root Users

A root user is a special user whose account can bypass access controls and other restrictions that might be enforced for regular users.

You can define multiple root users, each with their own set of credentials, to control access at a fine-grained level. For example, you can assign privileges to a user who needs root access for a particular task, but does not need the full set of root user privileges. Oracle Unified Directory enables you to configure each root user to have his own strong authentication mechanism (such as GSSAPI SASL), his own specific password policy, and his own resource limits.

A set of global root user privileges is defined by default. These privileges apply to all configured root users, including the default root user, unless you modify the privilege in the root user entry. You can change the global root user privileges that are inherited by all root users.

During the setup process, a default root user with full administrative rights is created. The DN proposed by the setup for this root user is `"cn=directory manager"`, so if you do not change the defaults proposed by the setup, a root user with the DN `"cn=directory manager,cn=Root DNs,cn=config"` is configured.

19.2.2 Configuring Root Users Using the Command-Line Utilities

You can view and edit the global root user properties by using command-line utilities.

To view and edit the global root user properties use the `dsconfig` command. To create and manage additional root users, you must use the `ldapmodify` command to add the user entries to the server configuration. The following sections describe how to manage root users by using the command line.

- Changing the Global Root User Privileges
- Creating a New Root User
- Editing an Existing Root User Using ldapmodify Command

19.2.2.1 Changing the Global Root User Privileges

You can change the global Root user privileges by using **dsconfig** command.

To display the global root user privileges, run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-root-dn-prop
Property                    : Value(s)
----------------------------:-------------------------------------------------
default-root-privilege-name : backend-backup, backend-restore, bypass-acl,
                            : bypass-lockdown, cancel-request, config-read,
                            : config-write, disconnect-client, ldif-export,
                            : ldif-import, modify-acl, password-reset,
                            : privilege-change, server-restart,
                            : server-shutdown, subentry-write,
                            : unindexed-search, update-schema
```

To change the global root user privileges, run the following dsconfig command run the `dsconfig set-root-dn-prop` command with the `--add` or `--remove` option.

The following example removes the default privilege of root users to perform a backup or restore operation on the server.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-root-dn-prop --remove default-root-privilege-name:backend-backup \
  --remove default-root-privilege-name:backend-restore
```

For a complete list of the privileges and an explanation of each privilege, see Understanding Privilege Subsystem.

19.2.2.2 Creating a New Root User

You can create a new Root user using an entry in an LDIF file.

To create a new root user, create the user entry in an LDIF file, then use the `ldapmodify` command to add the entry to the `cn=Root DNs,cn=config` branch in the server configuration.

> **Note:**
>
> The `cn=config` suffix is an administrative suffix and, as such, must be accessed using the administration connector. For more information see Managing Administration Traffic to the Server.

Suppose, for example, that you want to give a particular user the right to backup and restore a database, but no other administrative privileges.

1. Create an LDIF file that defines the root user entry with the correct privileges.

   The following sample LDIF file (`add-backup-admin.ldif`) defines a root user with the DN "`cn=backup-admin`" who has these privileges, but no other privileges on the server configuration.

```
dn: cn=backup-admin,cn=Root DNs,cn=config
changetype: add
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: ds-cfg-root-dn-user
objectClass: top
cn: backup-admin
sn: backup-admin
ds-cfg-alternate-bind-dn: cn=backup-admin
userPassword: secret
ds-privilege-name: backend-backup
ds-privilege-name: backend-restore
ds-privilege-name: -bypass-acl
ds-privilege-name: -bypass-lockdown
ds-privilege-name: -cancel-request
ds-privilege-name: -config-read
ds-privilege-name: -config-write
ds-privilege-name: -disconnect-client
ds-privilege-name: -ldif-export
ds-privilege-name: -ldif-import
ds-privilege-name: -modify-acl
ds-privilege-name: -password-reset
ds-privilege-name: -privilege-change
ds-privilege-name: -server-restart
ds-privilege-name: -server-shutdown
ds-privilege-name: -subentry-write
ds-privilege-name: -unindexed-search
ds-privilege-name: -update-schema
```

2. Use the ldapmodify command with the --useSSL option to add the LDIF file to the server configuration.

```
$ ldapmodify -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
  --useSSL -X -f add-backup-admin.ldif
```

For a complete list of the privileges and an explanation of each privilege, see Understanding Privilege Subsystem.

19.2.2.3 Editing an Existing Root User Using ldapmodify Command

You can edit an existing root user using **ldapmodify** command.

To edit an existing root user, use the `ldapmodify` command to change the attributes of the user entry under the `cn=Root DNs,cn=config` branch in the server configuration.

> **Note:**
>
> The `cn=config` suffix is an administrative suffix and, as such, must be accessed using the administration connector. For more information see Managing Administration Traffic to the Server.

The following example removes the capability of the root user created in the previous example to perform a restore operation.

```
$ ldapmodify -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
  --useSSL -X
dn: cn=backup-admin,cn=root DNs,cn=config
changetype: modify
delete: ds-privilege-name
ds-privilege-name: backend-restore
```

19.2.3 Configuring Root Users Using OUDSM

You can view and edit the default root user, and create and manage additional root users, by using the OUDSM interface.

The following topics describe how to configure root users by using OUDSM interface:

- Configuring the Global Root User Privileges
- Creating a New Root User
- Editing an Existing Root User Using OUDSM

19.2.3.1 Configuring the Global Root User Privileges

A set of global root user privileges is defined by default. These privileges apply to all configured root users, unless you modify the privilege in the root user entry.

To modify the global root user privileges by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Under the **General Configuration** item, select **Root Users**.

The global root user privileges are displayed in the right hand pane.

A check mark next to a privilege indicates that root users have that privilege by default.

4. To add a privilege to the list of global root user privileges, check the box next to that privilege.

   To remove a privilege, uncheck the box next to that privilege.

   For a complete list of the privileges and an explanation of each privilege, see Understanding Privilege Subsystem.

5. When you have made the modifications that you require, click **Apply**.

19.2.3.2 Creating a New Root User

You can create a new Root user using OUDSM.

To create a new root user by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. From the **Create** menu, select **Root User.**
4. In the **General Properties** region, enter the following details:
   a. In the **Name** field, type a name for the root user that you want to create.
   b. In the **Alternative Bind DNs** region, click **Add** to specify one or more alternative DNs that can be used when this root users binds to the server.

      For example, the alternative bind DN for the default root user is `"cn=Directory Manager"`. This allows you to bind as `"cn=Directory Manager"` instead of having to use `"cn=Directory Manager,cn=Root DNs,cn=config"`, which is the actual entry DN.

      The alternative bind DN must be unique among all root users.

      If you do not want to specify an alternative bind DN for the new root user, leave the table empty.

   c. In the **Password** field, enter a password for the root user.
   d. In the **Confirm Password** field, retype the password for the root user.
5. In the **Privileges** region, select the settings for the different privileges that must be applied to this new root user.

   For each privilege, you can select one of the following:

   - `Enable`. The privilege is enabled for this root user.
   - `Disable`. The privilege is disabled for this root user.
   - `Default Privilege (enable)` or `Default Privilege (disable`. The user inherits the default setting for this privilege, as defined in the global privilege configuration
6. Click **Create.**

   The following confirmation message appears:

   `Root User created successfully.`

19.2.3.3 Editing an Existing Root User Using OUDSM

You can edit an existing root user using OUDSM.

To edit an existing root user by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Under the **General Configuration** item, expand the **Root Users** item.
4. Select the root user whose configuration you want to change.

   The properties of the root user are displayed in the right hand pane.
5. Edit the required properties and click **Apply**.
6. You are prompted to save the new configuration. Click **Yes**.

19.3 Defining Groups

You can define groups using Oracle Unified Directory.Oracle Unified Directory supports *groups*, which are collections of entries that are manageable as a single object. Typically, directory administrators configure groups of printers, groups of software applications, groups of employees, and so forth.

Groups are especially useful when assigning special access privileges to a set of users. For example, you could configure a group of access managers and assign privileges that enables them to view confidential employee data, but restricts anyone else in the company from accessing that data.

The following group types are supported:

- Static groups. For more information, see Defining Static Groups.
- Dynamic groups. For more information, see Defining Dynamic Groups.
- Virtual static groups. For more information, see Defining Virtual Static Groups.

This section also describes about nested groups. For more information, see Defining Nested Groups.

19.3.1 Defining Static Groups

A static group defines its membership by providing explicit sets of distinguished names (DNs) using the `groupOfNames`, `groupOfUniqueNames`, or `groupOfEntries` object class. Static groups are well supported by external clients and provide good performance.

This section contains the following topics:

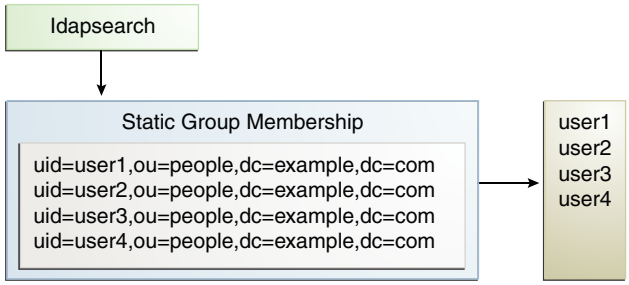
- Overview of Static Group
- Creating a Static Group With `groupOfNames`
- Creating a Static Group With `groupOfUniqueNames`
- Creating a Static Group With `groupOfEntries`
- Viewing All Members of a Static Group
- Viewing All Static Groups of Which a User Is a Member

- How to Find Whether a User is a Member of a Group

19.3.1.1 Overview of Static Group

A static group is one whose entry contains a membership list of explicit DNs. Many clients support static groups, but static groups are difficult to manage as the number of members in a group increases in size.

For example, if you have a member entry that requires a DN change, then you must change that user's DN for each group to which that user belongs.



The directory server supports the following three types of static groups, divided according to the object class they use:

- `groupOfNames`. You can define a static group by using the `groupOfNames` object class and by explicitly specifying the member DNs using the `member` attribute.

  > **Note:**
  >
  > RFC 4519 (`https://www.ietf.org/rfc/rfc4519.txt`) requires that the `member` attribute be mandatory within the `groupOfNames` object class. This membership requirement has traditionally caused data management problems when an administrator attempted to delete the last member in the group. The directory server solves this problem by allowing the `member` attribute to be optional. The optional membership requirement allows you to have an empty object class when you delete the last member of the group.

  ```
  dn: cn=Example Static Group 1,ou=Groups,dc=example,dc=com
  objectClass: top
  objectClass: groupOfNames
  member: uid=user1,ou=People,dc=example,dc=com
  member: uid=user2,ou=People,dc=example,dc=com
  cn: Example Static Group 1
  ```

- `groupOfUniqueNames`. You can define a static group by using the `groupOfUniqueNames` object class and by explicitly specifying the member DNs using the `uniqueMember` attribute. The `groupOfUniqueNames` object class differs from the `groupOfNames` object class in that you can enumerate the group's members by specifying a unique DN plus an optional identifier. The identifier ensures that the unique objects can be identified when adding, deleting, or renaming any object.

  For example, you could delete or move an employee (`cn=Tom Smith`) and add a new employee who has the same name (`cn=Tom Smith`) to the directory. To distinguish the two,

you must add a separate identifier by using a bit string. The following example shows two users with the same name, but the second `uniqueMember` has an optional identifier.

```
uniqueMember: uid=tsmith,ou=People,dc=example,dc=com
uniqueMember: uid=tsmith,ou=People,dc=example,dc=com#'0111101'B
```

> **Note:**
>
> Few LDAP applications actually use the optional UID identifier.
>
> RFC 4519 (https://www.ietf.org/rfc/rfc4519.txt) requires that the `uniqueMember` attribute be mandatory within the `groupOfUniqueNames` object class. This membership requirement has historically caused data management problems when an administrator tried to delete the last member in the group. Oracle Unified Directory solves this problem by allowing the `uniqueMember` attribute to be optional. The optional membership requirement allows you to have an empty object class when you delete the last member of the group.

```
dn: cn=Example Static Group 2,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
uniqueMember: uid=user1,ou=People,dc=example,dc=com
uniqueMember: uid=user2,ou=People,dc=example,dc=com
cn: Example Static Group 2
```

- `groupOfEntries`. You can define a static group using the `groupOfEntries` object class. Based on the original specifications (RFC 4519 (http://www.rfc-editor.org/rfc/rfc4519.txt) and `draft-findlay-ldap-groupofentries-00.txt`, which expired in March, 2008), the `groupOfEntries` object class differs from the `groupOfNames` and `groupOfUniqueNames` object classes in that attributes are optional, which enables you to specify an empty object class without any members.

> **Note:**
>
> Oracle Unified Directory supports the `groupOfEntries` draft but also allows empty `groupOfNames` and `groupOfUniqueNames` object classes. As a result, you can create empty groups of any type (`groupOfEntries`, `groupOfNames`, and `groupOfUniqueNames`).
>
> ```
> dn: cn=Example Static Group 3,ou=Groups,dc=example,dc=com
> objectClass: top
> objectClass: groupOfEntries
> cn: Example Static Group 3
> ```

19.3.1.2 Creating a Static Group With `groupOfNames`

This procedure describes how to reate a static group with `groupOfNames` command.

To create a static group with groupOfNames as object class:

1. Create the group entry in LDIF, including the group name (`cn`) and the `groupOfNames` object class.

   This example shows an LDIF file, named `static-group1.ldif`, that defines the new group.

```
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
cn: Directory Administrators
objectclass: top
objectclass: groupOfNames
ou: Groups
member: uid=ttully,ou=People,dc=example,dc=com
member: uid=charvey,ou=People,dc=example,dc=com
member: uid=rfisher,ou=People,dc=example,dc=com
```

2. Add the group by using `ldapmodify` to apply the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename static-group1.ldif
Processing ADD request for cn=Directory Administrators,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=Directory
Administrators,ou=Groups,dc=example,dc=com
```

3. Verify the change by using `ldapsearch` and the `isMemberOf` attribute.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --baseDN dc=example,dc=com "(uid=ttully)" isMemberOf
dn: uid=ttully,ou=People,dc=example,dc=com
isMemberOf: cn=Directory Administrators,ou=Groups,dc=example,dc=com
```

19.3.1.3 Creating a Static Group With `groupOfUniqueNames`

This procedure describes how to create a static group with `groupOfUniqueNames` as object class.

To create a static group with groupOfUniqueNames as object class:

1. Create the group entry in LDIF, including the group name (`cn`) and the `groupOfUniqueNames` object class.

   This example shows an LDIF file, named `static-group2.ldif`, that defines the new group.

```
dn: cn=Directory Administrators2,ou=Groups,dc=example,dc=com
cn: Directory Administrators2
objectclass: top
objectclass: groupOfUniqueNames
ou: Groups
uniquemember: uid=alangdon,ou=People,dc=example,dc=com
uniquemember: uid=drose,ou=People,dc=example,dc=com
uniquemember: uid=polfield,ou=People,dc=example,dc=com
```

2. Add the group by using `ldapmodify` to apply the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename static-group2.ldif
```

3. Verify the change by using `ldapsearch` and the `isMemberOf` attribute.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --baseDN dc=example,dc=com "(uid=rdaugherty)" isMemberOf
dn: uid=alangdon,ou=People,dc=example,dc=com
isMemberOf: cn=Directory Administrators2,ou=Groups,dc=example,dc=com
```

19.3.1.4 Creating a Static Group With `groupOfEntries`

This procedure describes how to create a static group with `groupOfEntries` as object class.

To create a static group with groupOfEntries as object class:

1. Create the group entry in LDIF, including the group name (`cn`) and the `groupOfEntries` object class.

   This example shows an LDIF file, named `static-group3.ldif`, that defines the new group.

   ```
   dn: cn=Directory Administrators3,ou=Groups,dc=example,dc=com
   cn: Directory Administrators3
   objectclass: top
   objectclass: groupOfEntries
   ou: Groups
   member: uid=bfrancis,ou=People,dc=example,dc=com
   member: uid=tjames,ou=People,dc=example,dc=com
   member: uid=bparker,ou=People,dc=example,dc=com
   ```

2. Add the group by using `ldapmodify` to apply the LDIF file.

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --defaultAdd --filename static-group3.ldif
   ```

3. Verify the change by using `ldapsearch` and the `isMemberOf` attribute.

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --baseDN dc=example,dc=com "(uid=bparker)" isMemberOf
   dn: uid=bparker,ou=People,dc=example,dc=com
   isMemberOf: cn=Directory Administrators3,ou=Groups,dc=example,dc=com
   ```

19.3.1.5 Viewing All Members of a Static Group

Use the `isMemberOf` virtual attribute to search for a group. The attribute is added to the user entry at the start of the search and then removed after the search has finished. This functionality provides easy management of groups with fast read access.

Use the `ldapsearch` command with the virtual attribute `isMemberOf`.

This example searches for all users who are members of the group "Accounting Managers."

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com \
  "(isMemberOf=cn=Accounting Managers,ou=Groups,dc=example,dc=com)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
ou: People
sn: Carter
facsimiletelephonenumber: +1 408 555 9751
roomnumber: 4612
userpassword: {SSHA}3KiJ51sx2Ug7DxZoq0vA9ZY6uaomevbJUBm7OA==
l: Sunnyvale
cn: Sam Carter
telephonenumber: +1 408 555 4798
givenname: Sam
uid: scarter
mail: scarter@example.com

dn: uid=tmorris,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
```

```
ou: People
sn: Morris
facsimiletelephonenumber: +1 408 555 8473
roomnumber: 4117
userpassword: {SSHA}bjFFHv6k1kbI6fZoCEfqmTj9XOZxWR06gxpKpQ==
l: Santa Clara
cn: Ted Morris
telephonenumber: +1 408 555 9187
givenname: Ted
uid: tmorris
mail: tmorris@example.com
```

19.3.1.6 Viewing All Static Groups of Which a User Is a Member

This procedure describes how to view all the static groups of which a user is a member of.

Search using `ldapsearch` and the virtual attribute `cn=IsMemberOf`, as shown in the following example:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com "(uid=scarter)" isMemberOf
dn: uid=scarter,ou=People,dc=example,dc=com
isMemberOf: cn=Accounting Managers,ou=groups,dc=example,dc=com
```

19.3.1.7 How to Find Whether a User is a Member of a Group

You can find out whether a user is a member of a group by using `ldapsearch` command.

Search using `ldapsearch`, as shown in the following example:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=Account Managers,ou=Groups,dc=example,dc=com" \
  "(&(objectclass=groupOfUniqueNames) \
  (uniquemember=uid=scarter,ou=People,dc=example,dc=com))"
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage accounting entries
cn: Accounting Managers
uniquemember: uid=scarter, ou=People, dc=example,dc=com
uniquemember: uid=tmorris, ou=People, dc=example,dc=com
```

19.3.2 Defining Dynamic Groups

A dynamic group defines its membership using a set of search criteria in the form of an LDAP URL, using the `groupOfUrls` object class. Dynamic groups handle large numbers of members well (millions of entries). As entries are updated, all parent groups are updated automatically.

A disadvantage of dynamic groups is that not all clients support them. Performance also is adversely affected if you must query the whole list of entries. Thus, dynamic groups are best suited for groups with a very large number of entries or for clients that need to determine specific group membership for an entry.

This section describes the following topics:

- Overview of Dynamic Group
- Creating a Dynamic Group
- Viewing All Members of a Dynamic Group

- Viewing All Dynamic Groups of Which a User Is a Member
- How to Find Whether a User Is a Member of a Dynamic Group

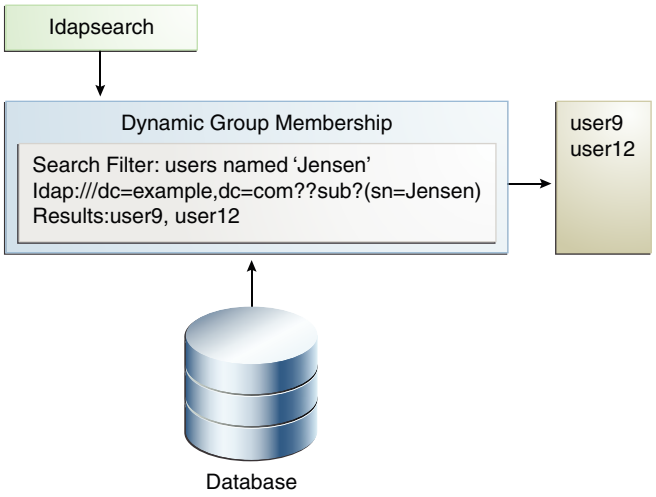
19.3.2.1 Overview of Dynamic Group

A *dynamic group* is one whose membership, rather than being maintained explicitly in a list, is determined by search criteria using an LDAP URL.

For example, suppose that you want to send an email to all managers in the `dc=example,dc=com` naming context. To do this, you create a dynamic group in which you specify `cn=Managers,ou=Groups,dc=example,dc=com`. You further specify that you want only email addresses returned. When the email application queries the directory for that particular group, the directory server computes the membership dynamically and returns the corresponding list of email addresses.

Dynamic groups use the `groupOfURLs` object class and the `memberURL` attribute to define LDAP URLs with the criteria (search base, scope, and filter) to be used for determining members of the group. The mechanism for determining whether a user is a member of a dynamic group is a constant-time operation, so it is just as efficient for groups with millions of members as it is for a group with only a few members. However, care must be taken when specifying the search criteria as it can adversely affect performance if searching over a large set of data.

Figure 19-1 Structure of a Dynamic Group



19.3.2.2 Creating a Dynamic Group

This procedure describes how to create a dynamic group.

To create a Dynamic Group:

1. Create an LDIF file that specifies the group.

   This example specifies the dynamic group for employees located at Cupertino.

```
dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
cn: CupertinoEmployees
objectclass: top
objectclass: groupOfURLs
```

```
ou: Groups
memberURL: ldap:///ou=People,dc=example,dc=com??sub?(l=Cupertino)
```

2. Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename dynamic_group.ldif
Processing ADD request for cn=cupertionEmployees,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=cupertionEmployees,ou=Groups,dc=example,dc=com
```

19.3.2.3 Viewing All Members of a Dynamic Group

This procedure illustrates the use of the virtual attribute `isMemberOf`. Do not use this procedure for very large groups, because it adversely affects the directory server's performance

Search using `ldapsearch` and the virtual attribute `isMemberOf`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "dc=example,dc=com" \
"(isMemberOf=cn=cupertinoEmployees,ou=Groups,dc=example,dc=com)"
dn: uid=abergin,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
...(more entries)...
```

19.3.2.4 Viewing All Dynamic Groups of Which a User Is a Member

This procedure describes how to view all dynamic groups of which a user is a member, using `ldapsearch` command.

Search using `ldapsearch` and the virtual attribute `isMemberOf`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com "(uid=abergin)" isMemberOf
dn: uid=abergin,ou=People,dc=example,dc=com
isMemberOf: cn=QA Managers,ou=groups,dc=example,dc=com
isMemberOf: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
```

19.3.2.5 How to Find Whether a User Is a Member of a Dynamic Group

Use `ldapsearch` command to find whether a user is a member of a dynamic group.

Search using `ldapsearch` and the virtual attribute `isMemberOf`, as shown below:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com \
"(&(uid=abergin)(isMemberOf=cn=cupertinoEmployees,ou=Groups,dc=example,dc=com))"
```

```
dn: uid=abergin,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
```

19.3.3 Defining Virtual Static Groups

A virtual static group appears and behaves like a static group to external clients, except that each member is represented by a virtual attribute that defines its membership as needed from another dynamic group.

The following topics describe virtual static groups:

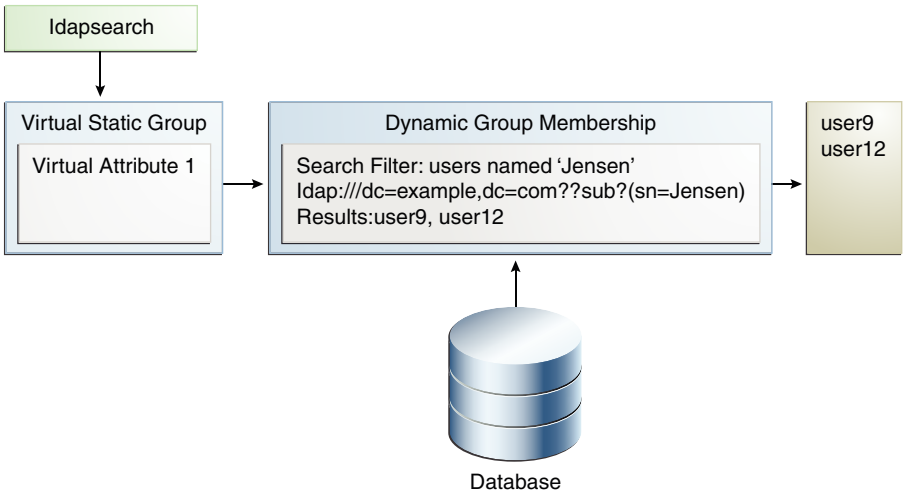
- Overview of Virtual Static Group
- Creating a Virtual Static Group
- Viewing All Members of a Virtual Static Group
- Viewing All Virtual Static Groups of Which a User Is a Member
- How to Find Whether a User is a Member of a Virtual Static Group

19.3.3.1 Overview of Virtual Static Group

A *virtual static group* allows access to dynamic groups by clients that can only support static groups.

In a virtual static group, each entry behaves like a static group entry by using virtual attributes. The virtual attributes are dynamically determined when invoked, and the operations that determine group membership are passed to another group, such as a dynamic group.

Figure 19-2 Virtual Static Group



Virtual static groups should include either the `groupOfNames` or `groupOfUniqueNames` object class but should not include the `member` or `uniqueMember` attribute. Virtual static groups should also contain the `ds-virtual-static-group` auxiliary object class and the `ds-target-group-dn` attribute. The `ds-target-group-dn` attribute is used to reference the actual group to mirror as a virtual static group and is used in place of the `member` or `uniquemember` attribute. For example:

```
dn: cn=Example Virtual Static Group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
objectClass: ds-virtual-static-group
cn: Example Virtual Static Group
ds-target-group-dn: cn=Example Real Group,ou=Groups,dc=example,dc=com
```

Virtual static groups are most efficient when the application issues a search targeted at the membership attribute but does not actually retrieve the entire set of members. It is common for applications to use a filter such as the following to attempt to determine whether a user is a member of a given group:

```
(&(objectClass=groupOfUniqueNames)(uniqueMember=uid=john.doe,\
  ou=People,dc=example,dc=com))
```

For applications that retrieve the set of members, virtual static groups might not be ideal because the process of constructing the entire member list can be expensive.

19.3.3.2 Creating a Virtual Static Group

This procedure describes how to create a virtual static group.

To create a Virtual Static Group:

1. Create an LDIF file that specifies the group.

   This sample file, `virtual-static.ldif`, specifies a virtual static group named `cupertinoEmployees`.

   ```
   dn: cn=virtualStatic,ou=Groups,dc=example,dc=com
   cn: Virtual Static
   objectclass: top
   objectclass: groupOfUniqueNames
   ```

```
objectclass: ds-virtual-static-group
ou: Groups
ds-target-group-dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
```

2. Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename virtual-static.ldif
Processing ADD request for cn=virtualStatic,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=virtualStatic,ou=Groups,dc=example,dc=com
```

19.3.3.3 Viewing All Members of a Virtual Static Group

Virtual static groups are best used in cases where the search is targeted at the membership attribute. This procedure is therefore not recommended but is included to show how to access the list.

This example uses the dynamic group, `cupertinoEmployees` that was created in the previous example.

Search using `ldapsearch` and the virtual attribute `cn=virtualStatic`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com "(isMemberOf=cn=virtualStatic,ou=Groups,dc=example,dc=com)"

dn: cn=virtualStatic,ou=Groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: ds-virtual-static-group
objectClass: top
cn: Virtual Static
uniqueMember: uid=abergin,ou=People,dc=example,dc=com
ds-target-group-dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
...(more entries)...
```

19.3.3.4 Viewing All Virtual Static Groups of Which a User Is a Member

This procedure describes how to view all the virtual static groups of which a user is a member, using `ldapsearch` command.

Search using `ldapsearch` and the virtual attribute `isMemberOf`:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b dc=example,dc=com "(uid=abergin)" isMemberOf
dn: uid=abergin,ou=People,dc=example,dc=com
isMemberOf: cn=QA Managers,ou=groups,dc=example,dc=com
isMemberOf: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
isMemberOf: cn=virtualStatic,ou=Groups,dc=example,dc=com
```

19.3.3.5 How to Find Whether a User is a Member of a Virtual Static Group

This procedure describes how to find whether a user is a member of a virtual static group.

Search using `ldapsearch` and the `uniqueMember` attribute.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=virtualStatic,ou=Groups,dc=example,dc=com" \
  "(&(objectclass=groupOfUniqueNames) \
  (uniquemember=uid=abergin,ou=People,dc=example,dc=com))"
dn: cn=virtualStatic,ou=Groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
objectClass: ds-virtual-static-group
ou: Groups
ds-target-group-dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
cn: Virtual Static
cn: virtualStatic
```

19.3.4 Defining Nested Groups

Groups can be nested, where one group is defined as a child group entry whose DN is listed within another group, its parent.

The following topics provide a conceptual description of nested group and the procedure how to create a nested group:

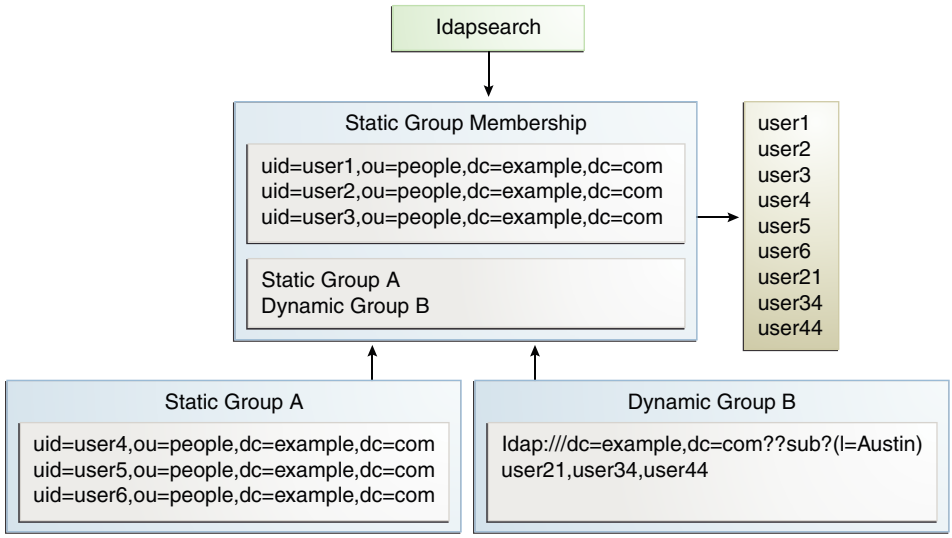
- About Nested Group
- Creating a Nested Group

19.3.4.1 About Nested Group

The nesting of groups allows you to set up inherited group memberships when performance is not a priority.

Groups can be nested, where one group is defined as a child group entry whose DN is listed within another group, its parent. The nesting of groups allows you to set up inherited group memberships when performance is not a priority. You can add zero or more member attributes with their values set to the DNs of nested child groups, including both static and dynamic groups.

Figure 19-3 Nested Static Group



19.3.4.2 Creating a Nested Group

This procedure describes how to create a nested group using one static group and one dynamic group.

To create a nested group using one static group and one dynamic group:

1. Create an LDIF file that specifies a static group.

   This example file, `static-group.ldif`, specifies a virtual static group named `Dev Contractors`.

   ```
   dn: cn=Contractors,ou=Groups,dc=example,dc=com
   cn: Dev Contractors
   objectclass: top
   objectclass: groupOfUniqueNames
   ou: Dev Contractors Static Group
   uniquemember: uid=wsmith,ou=Contractors,dc=example,dc=com
   uniquemember: uid=jstearn,ou=Contractors,dc=example,dc=com
   uniquemember: uid=pbrook,ou=Contractors,dc=example,dc=com
   uniquemember: uid=njohnson,ou=Contractors,dc=example,dc=com
   uniquemember: uid=sjones,ou=Contractors,dc=example,dc=com
   ```

2. Add the group by using `ldapmodify` to process the LDIF file.

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --defaultAdd --filename static-group.ldif
   ```

3. Create an LDIF file that specifies a dynamic group.

   This example file, `dynamic-group.ldif`, specifies a dynamic group named `Developers`.

   ```
   dn: cn=Developers,ou=Groups,dc=example,dc=com
   cn: Developers
   objectclass: top
   objectclass: groupOfURLs
   ou: Groups
   memberURL: ldap:///ou=People,dc=example,dc=com??sub?(ou=Product Development)
   ```

4. Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename dynamic-group.ldif
```

5. Create an LDIF file that specifies a nested static group.

   This example file, `nested-group.ldif`, specifies a nested group named `Developers Group`.

```
dn: cn=DevelopersGroup,ou=Groups,dc=example,dc=com
cn: Developers Group
objectclass: top
objectclass: groupOfUniqueNames
ou: Nested Static Group
uniquemember: cn=Contractors,ou=Groups,dc=example,dc=com
uniquemember: cn=Developers,ou=Groups,dc=example,dc=com
```

6. Add the group by using `ldapmodify` to process the LDIF file,

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --filename nested-group.ldif
```

19.4 Maintaining Referential Integrity

Referential integrity is a database mechanism for ensuring that all references are properly maintained after delete, rename, or move operations. For example, if an entry is removed from the directory, the directory server also removes the entry from any groups of which the entry is listed as a member.

The referential integrity mechanism is configured as a plug-in the directory server and can be enabled using the `dsconfig` command. For more information, see Managing the Server Configuration Using `dsconfig`.

The following topics describe about referential integrity plug-in:

- Overview of the Referential Integrity Plug-In
- Enabling the Referential Integrity Plug-In

19.4.1 Overview of the Referential Integrity Plug-In

By default, the referential integrity plug-in is disabled. When you enable the plug-in by using `dsconfig`, it performs integrity updates on the `member` and `uniquemember` attributes after a delete, rename, or move operation.

Referential integrity plug-in can be configured to do the processing either in background or in foreground mode based on `update-interval` configuration parameter. See “Referential Integrity Plugin” in *Configuration Reference for Oracle Unified Directory* for more information. By default, the value of `update-interval` is 0. If the `update-interval` value is 0, then the updates are made synchronously in the foreground. If `update-interval` value is set to > 0, then it runs in the background mode.

When referential integrity plug-in is configured to run in the background mode, if you delete, rename, or move a user or group entry in the directory, the operation is logged to the referential integrity log file, `INSTANCE_DIR/OUD/logs/referint`.

After a specified time, known as the `update-interval`, the server performs a search on the specified attributes and matches the results with the DNs of the deleted or modified entries recorded in the log. Then the server modifies the attributes based on the result of the match. The `referint` log is only updated when the value of `update-interval` is > 0. The DN log is

removed after the `update-interval` time elapses and then the user DN is removed from other entries based on the referential integrity plug-in parameter `config`.

You can configure the following referential integrity plug-in properties to suit your requirements:

- **Enabled**. Turn on the referential integrity plug-in.
- **plugin type**. By default, the delete, rename, and move operations are set. You can change a plug-in type to only delete, for example.
- **Attribute type**. By default, the attribute types are set to `member,uniquemember` but can be changed to some other attribute. If you use or define attributes containing DN values, you can use the referential integrity plug-in to monitor these attributes.
- **Base-DN**. By default, the scope is to use all public naming contexts but this can be changed to a specific context.
- **Log file**. By default, `logs/referint` is the log file. You can record the referential integrity updates in a different file. For example, if you want to record changes in a replicated environment, you can write to the *changelog* file on a replication server, so that it can be replicated to a consumer server.
- **Update-interval**. By default, the `update-interval` is set to 0 seconds, which will run referential integrity immediately after a delete, rename, or move operation. To minimize the impact of the updates on system performance, increase the amount of time between updates. Typical update intervals are as follows:
  - 0 seconds, update immediately
  - 90 seconds (updates every 90 seconds)
  - 3600 seconds (updates every hour)
  - 10,800 seconds (updates every 3 hours)
  - 28,800 seconds (updates every 8 hours)
  - 86,400 seconds (updates once a day)
  - 604,800 seconds (updates once a week)

19.4.2 Enabling the Referential Integrity Plug-In

To enable referential integrity by using `dsconfig`, set the `enabled` property of the plug-in to true.

This procedure describes how to enable the referential integrity plug-in:

as shown below:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-plugin-prop --plugin-name "Referential Integrity" --set enabled:true
```

19.5 Simulating ODSEE Roles in an Oracle Unified Directory Server

You can configure Oracle Unified Directory to simulate Oracle Directory Server Enterprise Edition (ODSEE) roles to satisfy grouping mechanism.

- About ODSEE Roles in an Oracle Unified Directory Server

19.5.1 About ODSEE Roles in an Oracle Unified Directory Server

Oracle Directory Server Enterprise Edition (ODSEE) includes a roles subsystem that is used to provide a specialized type of grouping mechanism. This capability is not included directly in Oracle Unified Directory, because it is based on nonstandard functionality, uses Netscape-proprietary schema elements, and is not widely used in LDAP-enabled applications.

However, Oracle Unified Directory does provide all of the functionality offered by ODSEE roles, and this functionality is available for use with standard grouping mechanisms. If you have an application that was specifically written to rely on the roles functionality available in ODSEE and cannot work with standard grouping mechanisms, you can configure Oracle Unified Directory to simulate ODSEE roles to satisfy such applications.

> **Note:**
>
> If your application needs to create and destroy role entries (for example, an entry containing one of the subordinates of the `nsRoleDefinition` object class), that functionality is currently not available in Oracle Unified Directory.

19.5.2 Determining Whether a User is a Member of a Role

If the application only needs to determine whether a user is a member of a given role, it should only need to look at the `nsRole` attribute in the target user's entry to determine whether the DN of the appropriate role is present. In this case, you can simulate role functionality by following the steps described in this section.

After completing these steps are completed, the `nsRole` virtual attribute appears as an operational attribute in user entries, and should include the DNs of all groups in which that user is a member.

> **Note:**
>
> The `nsRole` attribute is an operational attribute, and you must explicitly request that it returned in search results. You must also ensure that the authenticated user has permission to see that attribute.

1. Update the directory server to include the necessary schema for the ODSEE roles implementation.

   This schema is provided in the LDIF file, `03-dsee-roles.ldif`, as follows:

```
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License, Version 1.0 only
# (the "License").  You may not use this file except in compliance
# with the License.
#
```

```
# You can obtain a copy of the license at
# trunk/opends/resource/legal-notices/OpenDS.LICENSE
# or https://OpenDS.dev.java.net/OpenDS.LICENSE.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at
# trunk/opends/resource/legal-notices/OpenDS.LICENSE.  If applicable,
# add the following below this CDDL HEADER, with the fields enclosed
# by brackets "[]" replaced with your own identifying information:
#      Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# This file contains schema definitions required to simulate DSEE role
# functionality in OpenDS.
  dn: cn=schema
  objectClass: top
  objectClass: ldapSubentry
  objectClass: subschema
  attributeTypes: ( 2.16.840.1.113730.3.1.574 NAME 'nsRole'
  DESC 'Sun ONE defined attribute type' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  NO-USER-MODIFICATION USAGE directoryOperation
  X-ORIGIN 'Sun ONE Directory Server' )
  attributeTypes: ( 2.16.840.1.113730.3.1.575 NAME 'nsRoleDN'
  DESC 'Sun ONE defined attribute type' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  USAGE directoryOperation X-ORIGIN 'Sun ONE Directory Server' )
```

   a. Either copy the file into the `config/schema` directory of the directory server implementation and restart the server, or

   b. Use the `add schema file` task to cause the server to load the schema file into a running server instance.

2. Create a static or dynamic group to define role membership.

   Ensure that the group has an appropriate set of members.

3. Create a new instance of the `isMemberOf` virtual attribute to provide the `nsRole` virtual attribute.

   The `nsRole` attribute will include a list of the DNs of all groups in which the target user is a member. Use the `dsconfig` command to create the virtual attribute, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  create-virtual-attribute \
  --type is-member-of --name nsRole --set attribute-type:nsRole --set enabled:true
```

19.5.3 Altering Membership Using the `nsRoleDN` Attribute

If the application you are using expects to be able to alter membership by placing the name of the corresponding role in the `nsRoleDN` virtual attribute in a user's entry.

After these steps are completed, any user entry that contains an `nsRoleDN` value of `"cn=Test Role,ou=Roles,dc=example,dc=com"` also has that DN present in the `nsRole` operational attribute.

1. Create a dynamic group entry with the DN of the desired role.

2. Configure the group to include members that contain an `nsRoleDN` attribute with a value equal to the DN of the target role.

   For example, if the application is going to add an `nsRoleDN` value of "`cn=Test Role,ou=Roles,dc=example,dc=com`", add the following entry:

```
dn: cn=Test Role,ou=Roles,dc=example,dc=com
objectClass: top
objectClass: groupOfURLs
cn: Test Role
memberURL: ldap:///dc=example,dc=com??sub?(nsRoleDN=\
  cn=Test Role,ou=Roles,dc=example,dc=com)
```

Part IV

Configuring Proxy, Distribution, and Virtualization Functionality

This part describes how to configure proxy, distribution, and virtualization functionality for your deployments.

This section contains the following topics:

20
Configuring Access to Remote Data Sources

The data stored remotely can be accessed including identity data stored in an RDBMS. You can configure communication between a proxy instance and one or more remote LDAP servers.
The following topics describe configuring access to remote data sources:

- Configuring Access to Identity Data Stored in an RDBMS
- Configuring Communication With Remote LDAP Servers

20.1 Configuring Access to Identity Data Stored in an RDBMS

You can create a sample virtual configuration based on an Oracle Unified Directory proxy instance that exposes identity data stored in an Oracle Database as LDAP entries.

The examples in this section create this virtual view of the identity data by configuring an RDBMS workflow element and its supporting components. The RDBMS workflow element allows LDAP clients to access the identity data using the LDAP protocol.

This section includes the following topics:

- Understanding the RDBMS Workflow Element Use Case.
- Configuring the RDMBS Workflow Element.
- Creating the Components to Communicate with the RDBMS.
- About Granting Access to the Virtual Data.

For an overview of the RDBMS workflow element, see Enabling LDAP Clients to Access Identity Data Stored in an RDBMS.

> **Note:**
>
> The examples in this section use the `dsconfig` command to create and configure the RDBMS workflow element and other required components. The descriptions of these examples mention key options and properties you must set.
>
> For the description of all `dsconfig` subcommands and options, see dsconfig.

20.1.1 Understanding the RDBMS Workflow Element Use Case

The deployment of RDBMS workflow element use case includes LDAP clients, Oracle Unified Directory proxy servers, RDBMS workflow element and Oracle Database.

The deployment of RDBMS workflow element use case is described in the following topics:

- About LDAP Clients
- About Oracle Unified Directory Proxy Server

- About RDBMS Workflow Element and Supporting Components
- About Oracle Database

20.1.1.1 About LDAP Clients

This section introduces LDAP clients.

In this use case, LDAP clients want to access the identity data in an Oracle Database (the RDBMS) using the LDAP protocol. These clients do not want to execute SQL queries to access this data.

20.1.1.2 About Oracle Unified Directory Proxy Server

This use case requires an Oracle Unified Directory proxy server as the interface between the LDAP clients and the Oracle Database.

The proxy server connects to the Oracle Database as `dbuser`. The `dbuser` must have read privileges on the `PERSON`, `PHONE`, and `USER_GROUP` SQL tables in the Oracle Database, because LDAP searches are performed in the use case examples. If `dbuser` also wants to create or update the identity data using the LDAP protocol, then additional privileges are required.

> **Note:**
>
> Examples illustrated in the following sections assume that the schema contains the following tables only for demonstrational purpose: `PERSON`, `PHONE`, `USER_GROUP`. In a real deployment, actual tables from the schema are to be used.

20.1.1.3 About RDBMS Workflow Element and Supporting Components

The Oracle Unified Directory proxy uses the following components to communicate with the Oracle Database:

The proxy uses the following components to communicate with the Oracle Database:

- An **RDBMS extension** manages the connectivity with the remote Oracle Database through JDBC, by periodically checking the response from the remote peer and providing valid connections maintained by the connection pool.
- An **RDBMS workflow element** retrieves the connections from the RDBMS extension element, performs mapping between LDAP entries and SQL tables, and executes operations received from the LDAP clients.
- An **RDBMS workflow** for the RDBMS entries exposes the naming context handled by the RDBMS workflow element.
- An **access control group** for the RDBMS workflow uses **virtual ACIs** to control access the virtual identity data.

For more information, see Creating the Components to Communicate with the RDBMS.

20.1.1.4 About Oracle Database

The RDBMS in this use case is an Oracle 11*g* Database or later, which is installed, running, and populated with the deployment's identity data.

This database contains information about user accounts in these SQL tables:

- The `PERSON` table contains user data, including the employee ID, first name, last name, password, employee number, and hire date.
- The `PHONE` table, which is linked to the to the `PERSON` table, contains employee phone numbers.

For more information about these tables, see Understanding the Sample Schema for `PERSON` and `Phone` Tables.

This database also has the following characteristics:

- Database system identifier (SID): `orcl`
- Database URL: `myhost.example.com:1521:orcl`
- Database user who connects to the database from the proxy: `dbuser`
- Database user password: *`dbuser-password`*

In addition, if group memberships are needed, then those tables also have to be defined in the database. For more information, refer Understanding the Sample Schema for USER_GROUP Table.

20.1.2 Configuring the RDBMS Workflow Element

Before you begin configuring the RDBMS workflow element and its supporting components, you need to perform the required preliminary tasks.

The required preliminary tasks are:

- Setting Up an Oracle Unified Directory Proxy Server

20.1.2.1 Setting Up an Oracle Unified Directory Proxy Server

This use case requires an Oracle Unified Directory proxy server as the interface between the LDAP clients and the Oracle Database that contains the identity data.

To setup a proxy server instance using command-line tools on a UNIX or Linux system:

1. Ensure that your `JAVA_HOME` environment variable is set to a supported JVM installation.
2. Run the `oud-proxy-setup` script to set up the proxy server instance:

```
$ export INSTANCE_NAME=db-oud-proxy-instance
$ OUD_HOME/oud-proxy-setup --cli -p oud-port --adminConnectorPort admin-port
-D "cn=Directory Manager" -j password-file
```

In this example:

- `db-oud-proxy-instance` is the proxy instance directory name. This example sets the `INSTANCE_NAME` environment variable to this directory before running the `oud-proxy-setup` script.
- *`oud-port`* is the LDAP port used to access the proxy server instance.
- *`admin-port`* is the administration port.
- *`password-file`* contains the administrator password.

On Windows systems, run the `oud-proxy-setup.bat` script.

For more information, see Setting up Oracle Unified Directory as a Proxy Server in the *Installing Oracle Unified Directory*.

20.1.2.2 Installing a JDBC Driver JAR File for the RDBMS

The Oracle Unified Directory RDBMS implementation relies on the JDBC standard to communicate with the underlying RDBMS. If you are using Oracle Database, Oracle JDBC driver is already included and no action is needed. However, if you are using a non Oracle Database, you must install the JDBC driver JAR file that corresponds to the RDBMS you are using.

To install a JDBC driver JAR file:

1. Download the JDBC driver corresponding to the RDBMS database release you are using.
2. Copy the JAR file for the step 1 to the following directory:

   OUD_INSTANCE_NAME/OUD/lib
3. Restart the Oracle Unified Directory instance. For example, on UNIX and Linux systems:

   ```
   $ OUD_INSTANCE_NAME/OUD/bin/stop-ds
   $ OUD_INSTANCE_NAME/OUD/bin/start-ds
   ```

   On Windows systems, use `stop-ds.bat` and `start-ds.bat` in the *OUD_INSTANCE_NAME*\OUD\bat directory.

20.1.3 Creating the Components to Communicate with the RDBMS

You can create the components required for the Oracle Unified Directory proxy to communicate with the RDBMS.

Components required for communicating with the RDBMS is described below:

- Creating an RDBMS Extension
- Creating an RDBMS Extension to Use Secure Connection
- Creating an RDBMS Workflow Element
- Creating a Workflow for the RDBMS Entries
- Creating an Access Control Group for the RDBMS Workflow
- Associating the Workflow to a Network Group
- Configuring the LDAP-SQL Mappings

20.1.3.1 Creating an RDBMS Extension

An RDBMS extension corresponds to one RDBMS instance. This use case has only one Oracle Database instance.

To create an RDBMS extension named `ORCL1`, use the `dsconfig create-extension` command:

```
$ dsconfig create-extension \
--type rdbms \
--extension-name ORCL1 \
--set jdbc-driver-class:oracle.jdbc.driver.OracleDriver \
--set jdbc-url:"jdbc:oracle:thin:@myhost.example.com:1521:orcl" \
--set target-database:oracle14 \
--set rdbms-username:dbuser \
```

```
--set rdbms-password:dbuser-password \
--set enabled:true
```

In this example:

- `type` must be `rdbms` to specify an RDBMS extension.
- `extension-name` specifies the name of the new extension as `ORCL1`.
- `jdbc-driver-class` and `jdbc-url` correspond to the specific RDBMS instance.

  The URL depends on the host and port on which the RDBMS is running. The structure also depends on the specific RDBMS you are using. This example sets these properties for an 14*c* Oracle Database. For other databases, refer to the documentation for the JDBC driver you are using.
- `target-database` specifies the type of the RDBMS you are using. For a 14*c* Oracle Database, specify `oracle14`.
- `rdbms-username` and `rdbms-password` properties specify the credentials used to execute SQL queries.

  All SQL queries will be performed using these credentials, without consideration for the originating LDAP client identity. The virtual ACIs used to restrict access to the RDBMS SQL data based on the LDAP client identities will be configured later.

For a description of all RDBMS extension properties, see the *Oracle Fusion Middleware Configuration Reference for Oracle Unified Directory*.

20.1.3.2 Creating an RDBMS Extension to Use Secure Connection

You can configure an RDBMS extension to use a secure connection to access the database and prior to it, the database must be configured to accept secure connections. For the proxy to establish secure connections with remote RDMBS servers, you need to configure a truststore. An RDBMS proxy extension targeting a secured connection to a remote RDBMS data source must reference the appropriate truststore manager in its configuration.This reference enables the RDBMS proxy extension to access the imported remote RDBMS server certificate, to accept the secure connection.

The following topics describe creating an RDBMS extension to use secure connection:

- Configuring Security Between the Proxy and RDBMS Servers
- Creating an RDBMS Extension using `dsconfig`
- Configurable RDBMS Extension Properties Relevant to Security
- Configurable Advanced Properties for Setting Up JDBC Truststore and Keystore

20.1.3.2.1 Configuring Security Between the Proxy and RDBMS Servers

This task highlights the main steps required to configure security for connections to remote RDBMS servers. See *RDBMS Vendor's documentation* for detailed steps on configuring a keystore for remote RDBMS server connections and obtaining the RDBMS server certificate.

Perform the following steps to configure security between proxy and RDBMS server using `dsconfig`:

1. If the remote RDBMS server requires client authentication to be passed from the proxy, perform the following sub-steps. If the remote RDBMS server does not require client

authentication to be passed from the proxy, then proceed to Step 2 to configure a truststore.

a. Configure a keystore on the Oracle Unified Directory proxy server for remote RDBMS server connections.

   To do this, use the Java `keytool` command to generate a certificate on the proxy server. The keystore must be configured manually. See Configuring Key Manager Providers to configure a keystore manually.

   Self-sign the certificate or have the certificate signed by an external certificate authority. See Configuring Key Manager Providers for information about self-signing the certificate.

b. Configure a key manager provider on the proxy for the keystore for remote RDBMS server connections. See Configuring Key Manager Providers for configuring a key manager provider for remote RDBMS server. This key manager provider can be different to the one which is used for handling secure connections to clients.

c. See *RDBMS Vendor's documentation* for detailed steps to be followed to have an RDBMS server to trust the Oracle Unified Directory proxy certificate present in the above keystore during authentication.

2. For the proxy to establish secure connections with the remote RDBMS server, configure a truststore.

   An RDBMS proxy extension targeting a secured connection to a remote RDBMS data source must reference the appropriate truststore manager in its configuration. This reference enables the RDBMS proxy extension to access the imported remote RDBMS server certificate, to accept the secure connection.

3. Each truststore requires a proxy trust manager provider. For example:

   a. To list the proxy trust manager providers, use the `dsconfig list-trust-manager-providers` command. For example:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
   list-trust-manager-providers
   ```

   b. To create a proxy trust manager provider, use the `dsconfig create-trust-manager-provider` command. For example:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=Directory Manager"-j pwd-file-X -n \
   create-trust-manager-provider \
   --provider-name Backend\ Servers \
   --type file-based --set enabled:true \
   --set trust-store-file:/localhost/config/backend-servers-truststore \
   --set trust-store-type:JKS \
   --set trust-store-pin-file:/installPath/config/backend-servers-truststore.pin
   ```

4. Import the certificates of the remote RDBMS servers into the proxy truststore.

20.1.3.2.2 Creating an RDBMS Extension Using `dsconfig`

You can configure an RDBMS extension in Oracle Unified Directory using the `dsconfig` command to use a secure connection to access the database.

To create an RDBMS extension name `ORCL1` use `dsconfig create-extension command`:

```
$ dsconfig create-extension \
--type rdbms \
--extension-name ORCL1 \
--set jdbc-driver-class:oracle.jdbc.driver.OracleDriver \
--set jdbc-url:
```

```
'jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=%HOST NAME)(PORT=1522))
(CONNECT_DATA=(SERVICE_NAME=orcl12))(security=(ssl_server_cert_dn="CN=Root, C=US")))'
--set target-database:oracle \
--set rdbms-username:system \
--set rdbms-password:password \
--set enabled:true \
--set use-ssl:true \
--set ssl-trust-manager-provider:rdbms_truststore
```

See Table 20-1 for information about RDBMS Extension Properties.

If you already have an extension configured in a non-secured mode, you can switch to secured mode by running the following commands:

```
$ dsconfig set-extension-prop \
--extension-name ORCL1 \
--set use-ssl:true \
--set jdbc-url:'jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=%HOST NAME)
(PORT=1522))(CONNECT_DATA=(SERVICE_NAME=orcl12))(security=(ssl_server_cert_dn="CN=Root,
C=US")))'
--set ssl-trust-manager-provider:rdbms_truststore
```

Table 20-1lists the properties that need to be set as appropriate for secure connection.

The following example uses `set-extension-prop` to set TLS version and cipher suites using `jdbc-connection-properties` for Oracle database:

```
$ dsconfig set-extension-prop \
--extension-name ORCL1 \
--set jdbc-connection-properties:orcle.net.ssl_version=1.2 \

$ dsconfig set-extension-prop \
--extension-name ORCL1 \
--set jdbc-connection-
properties:oracle.net.ssl_cipher_suites=TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256
_CBC_SHA
```

> **Note:**
>
> Setting SSL protocol versions and cipher suites are not the same across different database vendors. Therefore, it is recommended that you refer to the corresponding database documentation to check how these properties can be set for the database over JDBC. RDBMS extension allows generic name-value pair properties to be set as per the database requirements. See, Configurable Advanced Properties for Setting Up JDBC Truststore and Keystore Details for more information on properties of JDBC truststore and keystore.

20.1.3.2.3 Configurable RDBMS Extension Properties Relevant to Security

You must set RDBMS Extension properties relevant to security using `dsconfig` to have a secured connection.

Table 20-1 lists the properties required to be set as appropriate for secure connection:

Table 20-1 RDBMS Extension Properties for Secured Connection

Property	Description
`type`	Must be `rdbms` to specify an RDBMS extension.
`extension-name` .	Specifies the name of the new extension as `ORCL1`
`jdbc-driver-class` and `JDBC URL`	Corresponds to the specific RDBMS instance.`JDBC URL` must be in URL form to activate a secure connection in the JDBC driver.
`target-database`	Specifies the type of the RDBMS you are using. For an 11g (and 12g) Oracle Database, specify Oracle11.
`rdbms-username` and `rdbms-password`	Specifies the credentials used to execute SQL queries.
`use-ssl`	Specifies a flag indicating a secure connection. The possible values are `true` or `false` based on the connection.
`ssl-trust-manager-provider`	Specifies the Trust Manager provider.
`ssl-key-manager-provider`	Specifies the Key Manager provider
`jdbc-connection-properties`	Lists the underlying JDBC driver properties as "name=value"
`jdbc-is-factory-properties`	Specifies flag indicating if the properties need to be set using `javax.net.setConnectionFactoryProperty` method. The possible values are `true` or `false`.

For a description of all RDBMS extension properties, see the *Configuration Reference for Oracle Unified Directory*.

20.1.3.2.4 Configurable Advanced Properties for Setting Up JDBC Truststore and Keystore Details

You must set the advanced properties to specify vendor-specific property name for truststore and keystore details. You must note that there are no specific JDBC standards for `jdbc-connection-properties`. Therefore, different vendors use different properties. Usually the database makes use of “javax.net.ssl.” prefix. Consequently the default values of all these properties are set to “javax.net.ssl.*”. Particularly, you do not require any additional configuration for **Oracle Database**.

If external non-Oracle databases do not use “javax.net.ssl.*” and instead use different names, for instance, IBM DB2 uses "sslTrustStoreLocation", then you must refer to *JDBC vendor documentation* and update the properties described as follows:

Table 20-2 Advanced Properties for Setting jdbc truststore and keystore

Property	Description
`jdbc-truststore-prop-name`	Specifies the vendor specific property name for specifying trust store, `javax.net.ssl.trustStore` is the default
`jdbc-truststore-pwd-prop-name`	Specifies the vendor specific property name for specifying trust store password,`javax.net.ssl.trustStorePassword` is the default
`jdbc-truststore-type-prop-name`	Specifies the vendor specific property name for specifying trust store type, `javax.net.ssl.trustStoreType` is the default

Table 20-2 (Cont.) Advanced Properties for Setting jdbc truststore and keystore

Property	Description
`jdbc-keystore-prop-name`	Specifies the vendor specific property name for specifying key store, `javax.net.ssl.keyStore` is the default
`jdbc-keystore-pwd-prop-name`	Specifies the vendor specific property name for specifying key store password. `javax.net.ssl.keyStorePassword` is the default
`jdbc-keystore-type-prop-name`	Specifies the vendor specific property name for specifying key store type. `javax.net.ssl.keyStoreType` is the default

For a description of all RDBMS extension properties, see *Configuration Reference for Oracle Unified Directory*.

20.1.3.3 Creating an RDBMS Workflow Element

You must create an RDBMS workflow element for each RDBMS extension you are using. You must also have configured an RDBMS extension before you create an RDBMS workflow element.

To create an RDBMS workflow element associated with the RDBMS extension you created in previous section, use the `dsconfig create-workflow-element` command:

```
$ dsconfig create-workflow-element \
--type rdbms \
--element-name ORCL1 \
--set rdbms-extension:ORCL1 \
--set suffix:o=db \
--set enabled:true
```

In this example:

- `type` must be `rdbms`, to specify an RDBMS workflow element.
- `element-name` specifies the name of the new RDBMS workflow element as `ORCL1`.
- `rdbms-extension` specifies the name of the extension associated with this workflow element as `ORCL1`.
- `suffix` specifies the suffix DN as `o=db`, which is the DN of all entries stored and exposed by this workflow element.

For a description of all RDBMS workflow element properties, see the *Oracle Fusion Middleware Configuration Reference for Oracle Unified Directory*.

20.1.3.4 Creating a Workflow for the RDBMS Entries

You must create a workflow to expose the naming context handled by the RDBMS workflow element. This workflow is defined by a naming context (base DN) and a workflow element that defines how Oracle Unified Directory should handle an incoming request.

To create a workflow associated with the RDBMS workflow element you previously created, use the `dsconfig create-workflow` command:

```
$ dsconfig create-workflow \
--workflow-name db \
```

```
--set base-dn:o=db \
--set workflow-element:ORCL1 \
--type generic \
--set enabled:true
```

In this example:

- `workflow-name` specifies the name of this configuration object as `db`.
- `base-dn` specifies the suffix associated with this workflow as `o=db`. This suffix must be the same as the `suffix` exposed by the RDBMS workflow element named `ORCL1`.
- `workflow-element` specifies the RDBMS workflow element as `ORCL1`.

For a description of all workflow properties, see the *Oracle Fusion Middleware Configuration Reference for Oracle Unified Directory*.

20.1.3.5 Creating an Access Control Group for the RDBMS Workflow

An RDBMS workflow is associated with an access control group that defines a list of ACIs that apply to the operations handled by the workflow.

To control access to the virtual directory view of data from the Oracle Database, you must enable and create virtual ACIs. When Oracle Unified Directory receives a request on a virtual directory data view, it uses the virtual ACIs and any authentication information provided by the user to allow or deny access to the requested data.

To create an access control group for the RDBMS workflow:

1. Create an access control group named `orcl1` using the `dsconfig create-access-control-group` command:

   ```
   $ dsconfig create-access-control-group \
   --group-name orcl1
   ```

   By default, the new access control group `orcl1` is empty, so at this point in the configuration, the virtual entries are exposed only to Oracle Unified Directory administrators.

2. Associate the access control group created in the previous step to the RDBMS workflow element using the `dsconfig set-workflow-prop` command:

   ```
   $ dsconfig set-workflow-prop \
   --workflow-name db \
   --set virtual-aci-mode:true \
   --set access-control-group:orcl1
   ```

In this example:

- `workflow-name` specifies that the workflow named `db` is protected by the virtual ACIs stored in the access control group named `orcl1`.
- `virtual-aci-mode` is set to `true`, so that all operations handling the ACI attribute manage this attribute as a virtual ACI. The attribute is no longer stored with user data. It is stored in the specific directory information tree (DIT) location `"cn=virtual acis"` in the Oracle Unified Directory proxy instance.

20.1.3.6 Associating the Workflow to a Network Group

Network groups are the single entry point of client requests to Oracle Unified Directory. A workflow must be registered with at least one network group, but it can be attached to several network groups.

To assign the `db` workflow to the default network group (`network-group`), use the `dsconfig set-network-group-prop` command:

```
$ dsconfig set-network-group-prop \
--group-name network-group \
--set workflow:db
```

You can now query the Oracle Unified Directory proxy to get the contents of the Oracle Database. By default, a dummy entry that corresponds to the base of the naming context exposed by the RDBMS workflow element is returned, since you have not configured the LDAP-SQL mappings yet.

To check your configuration, use the `ldapsearch` command:

```
$ ldapsearch -p oud-port -D "cn=directory manager" -w admin-password -b o=db
objectclass=*

dn : o=db
o : db
objectclass : organization
objectclass :top
```

20.1.3.7 Configuring the LDAP-SQL Mappings

You must now map the LDAP attributes to the appropriate columns in the SQL `PERSON`, `PHONE`, and `USER_GROUP` tables in the Oracle Database.

The following topics describe how to configure the LDAP-SQL mappings in this use case:

- Understanding the Sample Schema for `PERSON` and `Phone` Tables
- Creating RDBMS Tables
- Creating Object Class Mappings
- Creating Attribute Mappings
- Testing the Mappings
- Using Passwords Stored in the RDBMS
- Understanding the Sample Schema for `USER_GROUP` Table

20.1.3.7.1 Understanding the Sample Schema for `PERSON` and `Phone` Tables

In this use case, the Oracle Database exposes two SQL tables, a `PERSON` table containing user data, and a `PHONE` table containing user phone numbers.

The LDAP entries are mapped to the SQL rows and columns in these tables. One LDAP entry (`sqlPerson` object class) corresponds to each row of the `PERSON` table. The rows in the `PHONE` table appear in the multi-valued LDAP `telephoneNumber` attribute in the corresponding person entry.

The equivalent SQL commands to create these SQL tables are:

```
CREATE TABLE PERSON (ID INT PRIMARY KEY, FIRST_NAME VARCHAR(40), LAST_NAME
VARCHAR(40), PASSWORD VARCHAR(10), EMPLOYEE_ID VARCHAR(40), EMPLOYEE_NUMBER
INT, HIRE_DATE date)

CREATE TABLE PHONE (PERSON_ID INT, PHONE_NUMBER VARCHAR(17), FOREIGN
KEY(PERSON_ID) REFERENCES PERSON(ID) ON DELETE CASCADE, PRIMARY
KEY(PERSON_ID, PHONE_NUMBER))
```

In this use case, the primary keys for the `PERSON` table are automatically generated by the RDBMS and are not managed or exposed to the LDAP clients. This configuration is typical, because the LDAP entries are virtualized from the RDBMS and thus should be transparent to LDAP client applications.

In the Oracle Database, the primary key auto increment relies on the concept of database sequence and triggers. The following SQL commands create a sequence for the `PERSON` table and configure a trigger to automatically generate primary keys.

```
CREATE SEQUENCE PERSON_SEQUENCE START WITH 1 INCREMENT BY 1
CREATE OR REPLACE TRIGGER PERSON_TRIGGER BEFORE INSERT ON PERSON REFERENCING
NEW AS NEW FOR EACH ROW BEGIN SELECT PERSON_SEQUENCE.nextval
INTO :NEW.ID FROM dual; END;
```

20.1.3.7.2 Creating RDBMS Tables

You must create an RDBMS table object for each SQL table in the RDBMS that contains rows to be exposed as LDAP attributes and then associate these tables with an RDBMS workflow element.

To create the RDBMS tables for this use case, use the `dsconfig create-rdbms-table` command:

1. Create an RDBMS table named `PERSON` that corresponds to the SQL `PERSON` table in the Oracle Database and associate this table with the RDBMS workflow element named `ORCL1`:

   ```
   $ dsconfig create-rdbms-table \
     --set db-table-name:PERSON \
     --table-name PERSON \
     --element-name ORCL1 \
     --set primary-key-field:ID \
     --set primary-key-storability:false \
     --set db-sequence-name:PERSON_SEQUENCE \
     --type generic
   ```

   In this example:

   - `table-name` specifies the name of the table configuration object.
   - `db-table-name` specifies the name of the corresponding SQL table.
   - `element-name` specifies the name of the RDBMS workflow element to which this table is associated with.
   - `primary-keyfield` specifies the SQL column (or columns) that correspond to the SQL primary keys.
   - `primary-key-storability` specifies whether the primary key of this table can contain user-provided values. If set to `true`, key values can be inserted into this column by end-users

     This use case uses auto-generated primary keys, so `primary-key-storability` is set to `false`. In most deployments, the key management should be transparent to LDAP

clients, so that the key is automatically generated by the RDBMS when a row is inserted.

- `db-sequence-name` specifies the database sequence to generate the primary key field value when database sequences are used with triggers. This is the case for an Oracle Database.

2. Create an RDBMS table named `PHONE` that corresponds to the SQL `PHONE` table in the Oracle Database and associate the table with the RDBMS workflow element named `ORCL1`:

```
$ dsconfig create-rdbms-table \
  --table-name PHONE \
  --element-name ORCL1 \
  --set db-table-name:PHONE \
  --set primary-key-field:PERSON_ID \
  --set primary-key-field:PHONE_NUMBER \
  --set cascade-delete-on-relation:true \
  --set join-type:many-to-one \
  --set join-rule:PHONE.PERSON_ID=PERSON.ID \
  --type generic
```

For a description of all RDBMS table properties, see the *Oracle Fusion Middleware Configuration Reference for Oracle Unified Directory*.

20.1.3.7.3 Creating Object Class Mappings

An object class mapping configuration object specifies the name of the LDAP object class that corresponds to the LDAP objects built from the SQL table content. If the object class is not defined in the server schema, it is added automatically to the server schema during server startup.

To create an object class mapping for the `sqlPerson` object class in the RDBMS workflow element, use the `dsconfig create-objectclass-mapping` command:

```
$ dsconfig create-objectclass-mapping \
  --mapping-name sqlPerson \
  --element-name ORCL1 \
  --set objectclass-name:sqlPerson \
  --set rdn-attribute:uid \
  --type generic
```

In this example:

- `objectclass-name` specifies the name of the LDAP object class that will appear in the `objectclass` attribute.
- `rdn-attribute` specifies `uid` as the LDAP attribute used as a naming attribute.

  In this use case, the `uid` attribute corresponds to the `EMPLOYEE_ID` column in the `PERSON` table, as shown in the next section.

20.1.3.7.4 Creating Attribute Mappings

You must create an attribute mapping configuration object for each SQL row exposed as a LDAP attribute.

The attribute mappings required for this use case are shown in the following table:

LDAP Attribute	SQL Table and Column
`uid`	`PERSON:EMPLOYEE_ID`

LDAP Attribute	SQL Table and Column
lastName	PERSON:LAST_NAME
firstName	PERSON:FIRST_NAME
employeeNumber	PERSON:EMPLOYEE_NUMBER
hireDate	PERSON:HIRE_DATE
userPassword	PERSON:PASSWORD
telephoneNumber	PHONE:PHONE_NUMBER

To create attribute mappings for the `sqlPerson` object class, use the `dsconfig create-attribute-mapping` command for each attribute shown in the previous table:

```
$ dsconfig create-attribute-mapping \
  --attribute-mapping-name employeeID \
  --mapping-name sqlPerson \
  --set attribute-name:uid \
  --set field-name:EMPLOYEE_ID \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic

$ dsconfig create-attribute-mapping \
  --attribute-mapping-name firstName \
  --mapping-name sqlPerson \
  --set attribute-name:firstName
  --set field-name:FIRST_NAME \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic

$ dsconfig create-attribute-mapping \
  --attribute-mapping-name lastName \
  --mapping-name sqlPerson \
  --set attribute-name:lastName \
  --set field-name:LAST_NAME \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic

$ dsconfig create-attribute-mapping \
  --attribute-mapping-name employeeNumber \
  --mapping-name sqlPerson \
  --set attribute-name:employeeNumber \
  --set field-name:EMPLOYEE_NUMBER \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic

$ dsconfig create-attribute-mapping \
  --attribute-mapping-name hireDate \
  --mapping-name sqlPerson \
  --set attribute-name:hireDate \
  --set field-name:HIRE_DATE \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic

$ dsconfig create-attribute-mapping \
```

```
  --attribute-mapping-name telephoneNumber \
  --mapping-name sqlPerson \
  --set attribute-name:telephoneNumber \
  --set field-name:PHONE_NUMBER \
  --set table-name:PHONE \
  --element-name ORCL1 \
  --type generic
```

In these examples:

- `attribute-mapping-name` specifies a name for the mapping performed by this command.
- `attribute-name` specifies the LDAP attribute being mapped to the indicated SQL table and column.
- `mapping-name` specifies the object class.
- `field-name` and `table-name` specify the SQL column and table names for mapping the LDAP attribute.

20.1.3.7.5 Testing the Mappings

At this stage of the configuration, each row of the `PERSON` table is exposed as an instance of `sqlPerson` in the `o=db` suffix. The corresponding `telephoneNumber` (if it exists) is retrieved from the `PHONE` table.

To test these mappings, perform an LDAP search, as shown in the following example.

> **Note:**
>
> You must perform the following test as Directory Manager, because at this point, there are no ACIs granting access to the RDBMS workflow element content.

```
$ ldapsearch -p oud-port -D "cn=directory manager" -w admin-password -b  o=db
objectclass=*

dn : o=db
o : db
objectclass : organizationalUnit
objectclass :top
dn : uid=53422345,o=db
objectclass : sqlPerson
objectclass :top
uid : 53422345
firstName : Joseph
lastName : Smith
employeeNumber : 172453
hireDate : 199505011220000.000Z
telephoneNumber : +33123456789

...
```

The previous example shows the first entry returned by the `ldapsearch` command. If the mappings are configured correctly, the search returns virtual LDAP entries built from the SQL tables, according to the defined mapping rules.

20.1.3.7.6 Using Passwords Stored in the RDBMS

In the Oracle Database, the `PASSWORD` column in the `PERSON` table contains the user password.

To configure the RDBMS workflow element to allow LDAP clients to authenticate against the password stored in the Oracle Database:

1. Create an attribute mapping for the `userPassword` attribute using the `dsconfig create-attribute-mapping` command:

```
$ dsconfig create-attribute-mapping \
  --attribute-mapping-name userPassword \
  --mapping-name sqlPerson \
  --set attribute-name:userPassword \
  --set field-name:PASSWORD \
  --set table-name:PERSON \
  --element-name ORCL1 \
  --type generic
```

   In this example:

   - `attribute-name` specifies `userPassword` as the LDAP attribute to map.
   - `field-name` and `table-name` specify the SQL column and table names for the mapping.
   - `element-name` specifies the `ORCL1` RDBMS workflow element.

2. Configure the RDBMS workflow element using the `dsconfig set-workflow-element-prop` command, so that the workflow element can use the `userPassword` attribute for authentication:

```
$ dsconfig set-workflow-element-prop \
  --element-name ORCL1 \
  --set password-attribute:userpassword \
  --set password-storage-scheme:"Salted SHA-512"
```

In this example:

- `password-attribute` specifies the attribute that contains the user password.
- `password-storage-scheme` specifies how the user password is stored in the Oracle Database.

  In this example, the user password is stored in the Oracle Database hashed using the `Salted SHA-512` algorithm. Unlike in LDAP entries, hashed password values in SQL tables are not prefixed by the digest algorithm tag (such as `{SSHA-512}`).

20.1.3.7.7 Understanding the Sample Schema for `USER_GROUP` Table

> **Note:**
>
> If group membership details are stored across multiple tables, then you need to define the appropriate joins in the configurations.

In this section, the example illustrated is only for demonstrational purpose. In a real deployment scenario, you need to perform all configurations (rdbms table creation, objectclass mappings, attribute mappings) based on the actual tables.

In this use case, consider the following table:

```
CREATE TABLE USER_GROUP(ID NUMBER NOT NULL, GROUP_NAME VARCHAR2(10),
GROUP_MEMBER VARCHAR2(40));
```

where GROUP_NAME column holds the group's name, while GROUP_MEMBER column holds the name of the member of the Group as follows:

Table 20-3 Group Name Table

ID	GROUP_NAME	GROUP_MEMBER
1234	Group 1	user1
1235	Group 1	user2
1236	Group 1	user3

Create RDBMS Table, Object class mappings and Attribute Mappings by referring the respective sections: Creating RDBMS Tables, Creating Object Class Mappings, and Creating Attribute Mappings .

For example,

```
$ dsconfig create-rdbms-table \
--set db-table-name:USER_GROUP \
--table-name USER_GROUP \
--element-name ORCL1 \
--set primary-key-field:ID \
--set primary-key-storability:false \
--set db-sequence-name:USER_GROUP_SEQUENCE \
--type generic


$ dsconfig create-objectclass-mapping \
--mapping-name sqlGroup \
--element-name ORCL1 \
--set objectclass-name:groupOfUniqueNames \
--set rdn-attribute:cn \
--type generic


$ dsconfig create-attribute-mapping \
--attribute-mapping-name groupCN \
--mapping-name sqlGroup \
--set attribute-name:cn \
--set field-name:GROUP_NAME \
--set table-name:USER_GROUP \
--element-name ORCL1 \
--type generic


$ dsconfig create-attribute-mapping \
```

```
--attribute-mapping-name groupUniquemember \
--mapping-name sqlGroup \
--set attribute-name:uniquemember \
--set field-name:GROUP_MEMBER \
--set table-name:USER_GROUP \
--element-name ORCL1 \
--type generic
```

> **Note:**
>
> To return the User's DN as uniquemember attribute values in the ldap search results for Groups, a Transformation should be specified.

A sample Transformation configuration is provided here. For more information, refer Understanding the Transformation Framework.

- Create transformation for attribute uniquemember:

```
dsconfig create-transformation \
--type map-attribute \
--transformation-name uniquemember_transform \
--set client-attribute:uniquemember="{uid=%uniquemember%,o=db}" \
--portProtocol LDAP
```

- Create workflow element for transformation:

```
dsconfig create-workflow-element \
--set next-workflow-element:ORCL1 \
--set transformation:uniquemember_transform  \
--portProtocol LDAP \
--type transformations \
--element-name uniquemem_transform_wfe \
--set enabled:true

dsconfig set-workflow-element-prop \
--element-name uniquemem_transform_wfe \
--add transformation:uniquemember_transform
```

- Create workflow for transformation:

```
dsconfig create-workflow \
--set base-dn:o=db \
--set enabled:true \
--set workflow-element:uniquemem_transform_wfe  \
--workflow-name workflow1 \
--type generic
```

- Set Network Group for transformation:

```
dsconfig set-network-group-prop \
--group-name network-group \
--set workflow:workflow1
```

With the above configurations in place, LDAP search result is as follows:

```
dn: cn=group1,o=db
uniquemember: cn=user1,o=db
uniquemember: cn=user2,o=db
uniquemember: cn=user3,o=db
cn: group1
objectClass: top
objectClass: groupOfUniqueNames
```

20.1.4 About Granting Access to the Virtual Data

You can grant access to the virtual data by creating virtual ACI's through ldap commands.

> **Note:**
>
> Your access control strategy for the RDBMS you are using depends on your corporate policies, so you must create virtual ACIs to follow those policies.

By default, the access control group named `orcl1` created in Creating an Access Control Group for the RDBMS Workflow is empty. The virtual entries for the database are exposed only to Oracle Unified Directory administrators. Thus, the following search does not return any entries:

```
$ ldapsearch -p oud-port -D uid=53422345,o=db -w password -b o=db objectclass=*
```

The following command creates a virtual ACI granting full access to the owner of the virtual entry created from the Oracle Database:

```
ldapmodify -p oud-port -D "cn=Directory Manager" -w admin-password
dn : o=db
changetype : modify
add : aci
aci : (targetattr= "*") (version 3.0 ; acl "self example" ; allow (all) userdn="ldap:///
self" ;)
```

If you retry the previous search using this new virtual ACI, each user is granted access to their own entry based on the `uid`:

```
ldapsearch -p oud-port -D uid=53422345,o=db -w password -b o=db objectclass=*

dn : uid=53422345,o=db
objectclass : sqlPerson
objectclass :top
uid : 53422345
firstName : Audrey
lastName : Smith
employeeNumber : 172453
hireDate : 199505011220000.000Z
telephoneNumber : +33123456789
```

20.2 Configuring Communication With Remote LDAP Servers

Configuring communication between a proxy instance and one or more remote LDAP servers and the tasks associated with this is described in the following sections.

The following topics describe the communication with remote LDAP:

- Configuring LDAP Server Extensions
- Configuring Proxy LDAP Workflow Elements
- Configuring the Bind Mode

> **Note:**
>
> For more information about communicating with remote LDAP servers, see Understanding How to Enable Communication with a Remote LDAP Server.

20.2.1 Configuring LDAP Server Extensions

LDAP server extensions can be configured to communicate with the remote LDAP servers.

The following topics describe the tasks involved in configuring LDAP server extensions:

- Viewing the Existing LDAP Server Extensions
- Viewing LDAP Server Extension Properties
- Viewing Advanced LDAP Server Extension Properties
- Creating an LDAP Server Extension
- Modifying the Properties of an LDAP Server Extension
- Modifying the Advanced Properties of an LDAP Server Extension
- Modifying the LDAP Data Source Monitoring Connection Properties

20.2.1.1 Viewing the Existing LDAP Server Extensions

This section describes the procedure to view the existing LDAP server extensions.

To view a list of all the LDAP server extensions configured for a proxy instance, use the `dsconfig list-extensions` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-extensions

Extension  : Type
-----------:--------------------
gi-catalog : global-index-catalog
proxy1     : ldap-server
proxy2     : ldap-server
```

The extensions with type `ldap-server` are the LDAP server extensions. You should have one LDAP server extension for each remote LDAP server.

20.2.1.2 Viewing LDAP Server Extension Properties

You can view the LDAP server extension properties using the `dsconfig` command.

To view the properties of a specific LDAP server extension, use the `dsconfig get-extension-prop` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-extension-prop --extension-name proxy1

Property                  : Value(s)
--------------------------:---------
enabled                   : true
remote-ldap-server-address : server1.example.com
remote-ldap-server-port    : 1389
page-size                 : 0
```

The following properties are displayed:

- enabled

  Indicates if the LDAP server extension is enabled (`true`) or not (`false`)

- remote-ldap-server-address and remote-ldap-server-port

  Indicates the address and port of the remote LDAP server to which requests will be forwarded

- monitoring-bind-dn and monitoring-bind-password

  These properties are displayed only if you specify the `--advanced` option. They provide the credentials of the user that the extension will use to perform monitoring of the data source. If these properties have not been changed from the default, then they are not displayed. Monitoring is then performed anonymously.

  To configure these properties, see Monitoring the Server with LDAP.

- page-size

  Indicates the default page size to be considered when requesting entries from the Directory Server backend associated with this LDAP Server Extension.

  This restricts the number of entries (when configured > 0) that can be requested from the backend server to be configured with this extension. It is especially useful when clients while accessing OUD Proxy use a search query that does not explicitly contain the paged search option, as the search may timeout if the query results are quite large.

  This option should have a value less than or equal to the size limit parameter specified by the backend to avoid erroneous results.

  This parameter is honored only when the client does not provide the page size search option in its request.

  The default value is `0`.

20.2.1.3 Viewing Advanced LDAP Server Extension Properties

You can view all of the LDAP server extension properties using `dsconfig` command.

To view all of the LDAP server extension properties, use the `dsconfig --advanced get-extension-prop` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  --advanced get-extension-prop --extension-name proxy1
```

Properties similar to the following are displayed.

```
         Property                          Value(s)
        ------------------------------------------------------------------------
    1)  enabled                           true
    2)  java-class                        com.sun.dps.server.workflowelement
                                          .proxyldap.LDAPServerExtension
    3)  monitoring-check-interval         30000
    4)  monitoring-connect-timeout        5000
    5)  monitoring-inactivity-timeout     120000
    6)  monitoring-ping-timeout           5000
    7)  page-size                         0
    8)  pool-increment                    5
    9)  pool-initial-size                 10
    10) pool-max-size                     1000
    11) pool-max-write                    0
    12) pool-release-connection-interval  300000
    13) pool-use-max-write                false
    14) proxied-auth-use-v1               false
    15) remote-ldap-server-address        localhost
    16) remote-ldap-server-connect-timeout 10000
    17) remote-ldap-server-port           1389
    18) remote-ldap-server-read-only      false
    19) remote-ldap-server-read-timeout   10000
    20) remote-ldap-server-ssl-policy     never
    21) remote-ldap-server-ssl-port       636
    22) saturation-precision              5
    23) ssl-client-alias                  -
    24) ssl-key-manager-provider          -
    25) ssl-trust-all                     false
    26) ssl-trust-manager-provider        -
    27) ssl-protocol                  system default value
    28) ssl-cipher-suite                  system default value
```

> **Note:**
>
> Most of the advanced properties (except SSL properties) are set by default when the LDAP server extensions are created.

To modify these values, see Modifying the Properties of an LDAP Server Extension.

To understand about the monitoring properties, see Modifying the LDAP Data Source Monitoring Connection Properties.

To understand about the SSL (security) properties, see Configuring Security Between the Proxy and the Data Source.

To understand about system default values, see Supported TLS Protocols and Cipher Suites by Oracle Unified Directory

20.2.1.4 Creating an LDAP Server Extension

You can create an LDAP server extension using `dsconfig` command.

To create a new LDAP server extension, use the `dsconfig create-extension` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-extension \
  --extension-name DS-proxy5 \
  --type ldap-server \
  --set enabled:true \
  --set remote-ldap-server-address:DS5-hostname
  --set remote-ldap-server-port:1389
```

The extension `type` must be `ldap-server`. The name of the new extension is defined by `extension-name`, in this example `DS-proxy5`.

You must also specify the name of the remote LDAP server with which this extension is associated (`remote-ldap-server-address`). You can specify either the hostname or the IP address of the remote LDAP server.

If you do not specify a `remote-ldap-server-port`, the default LDAP port of 1389 is assumed.

20.2.1.5 Modifying the Properties of an LDAP Server Extension

This section describes the procedure to modify the properties of an LDAP server extension.

To modify the LDAP server extension properties, use the `set-extension-prop` subcommand. This subcommand enables you to do the following:

- Set whether the LDAP server extension is enabled (`true`) or not (`false`)
- Modify the remote LDAP directory server address and port (`remote-ldap-server-address` and `remote-ldap-server-port`)
- Set the credentials of the user that the extension will use to perform monitoring of the data source (`monitoring-bind-dn` and `monitoring-bind-password`). If left blank, the monitoring will be performed anonymously, which is the default.

For example, changing the remote LDAP server is a common operation. You must set the new remote LDAP server address and port, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-extension-prop \
  --extension-name DS-proxy5 \
  --set remote-ldap-server-address:DS5-hostname \
  --set remote-ldap-server-port:3388
```

To modify advanced LDAP server extension properties, see Modifying the Advanced Properties of an LDAP Server Extension.

20.2.1.6 Modifying the Advanced Properties of an LDAP Server Extension

This section lists the advanced properties that you can configure.

You can configure the following advanced properties:

- **pool-increment**

The increment by which the size of a connection pool is increased or decreased. If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the incremental change in size of each pool is set to `pool-increment`.

The default value is 5 connections.

- **pool-initial-size**

  The initial size of a connection pool. This is the initial number of connections to be created when a pool is initialized. The `pool-initial-size` property is also the minimum size of a pool.

  The default value is 10 connections.

  If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the initial size, and minimum size, of each pool is set to `pool-initial-size`. Therefore there can initially be twice the total number of connections indicated in `pool-initial-size`.

  For more information, see Understanding the Modes of Secure Connection.

- **pool-max-size**

  The maximum size of a connection pool. This is the maximum number of connections that a pool can allocate. If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the maximum size of each pool is set to `pool-max-size`.

  The default value is 1000 connections.

- **pool-max-write**

  The maximum number of write connections that a connection pool can allocate at the same time. This is an integer. This parameter is taken into account only if the `pool-use-max-write` parameter is set to `true`.

  The default value is 0 connections.

- **pool-release-connection-interval**

  The time after which a connection is considered by the proxy to be unused if no traffic has been sent on it. This reduces the size of the pool of connections, if the pool has been previously increased. If the number of unused connections is greater than `pool-increment`, then the size of the pool is reduced by `pool-increment`. This means that unused connections are closed and are removed from the pool.

  The default value is 300000 milliseconds (30 seconds).

- **pool-use-max-write**

  If this boolean is set to true, the `pool-max-write` parameter is taken into account, otherwise it is not.

  By default, `pool-use-max-write` is set to false.

- **proxied_auth_use_v1**

  When using the proxy authorization control mode, the default version of the control is v2. To use an older version for compatibility reasons, set `proxied-auth-use-v1` to true.

  By default, `proxied-auth-use-v1` is set to false.

  For more information about controls, see LDAP Controls and Operations Reference.

- **remote-ldap-server -read-timeout**

The timeout for reads. If the timeout is reached before the remote LDAP server sends back a response, an error is returned by the proxy to the client.

By default, this value is 10000 milliseconds (10 seconds).

- **saturation-precision**

  The saturation precision is used in calculating the saturation threshold. Since the saturation limit can vary as requests are sent and received, the saturation precision indicates how much change the saturation should get before the saturation is taken into account.

  By default the saturation can vary by 5% before it is taken into account.

- **ssl-protocol**

  Governs the SSL/TLS protocol that would be used during SSL communication with the remote LDAP server. This property takes system default values and can be overridden with valid SSL/TLS protocols as required. See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to understand about system default values.

- **ssl-cipher-suite**

  Governs the SSL/TLS cipher suite that would be used during SSL communication with the remote LDAP server. This property takes system default values and can be overridden with valid SSL/TLS cipher suites as required. See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to understand about system default values.

- **allow-server-supported-controls**

  If the `allow-server-supported-controls` property is set to `true`, the LDAP Server Extension will fetch all the controls set in `supportedControl` of the remote backend's RootDSE during initialization. It will pass only those controls to the remote server and will ignore and not send any other LDAP controls to the remote backend in an LDAP request. However, if this property is set to `false`, then the LDAP Server Extension will honor values set in the `ignored-ldap-controls` property.

- **ignored-ldap-controls**

  A list of OIDs of LDAP controls which will be ignored and not sent to the remote backend.

- **remote-ldap-server-guid**

  Specifies the name of the attribute that stores LDAP entry's global unique identifier (GUID) value in the remote LDAP server. For example, in case of Active Directory, it is `objectGUID`.

- **server-guid-name**

  Specifies the attribute name, which will contain the transformed value of `remote-ldap-server-guid` in the entries returned by the LDAP Server extension. The default value is `orclguid`.

- **remote-ldap-server-additional-guids**

  Specifies the names of the attributes other than what is configured in `remote-ldap-server-guid` property, which may be storing GUID values in the remote LDAP server. When configured, the attribute value is transformed into `orclguid` format by the LDAP Server extension.

The monitoring properties are described in Modifying the LDAP Data Source Monitoring Connection Properties.

The SSL properties are security features. For information about these properties, see Configuring Security Between the Proxy and the Data Source.

To modify the advanced LDAP server extension properties, use the `set-extension-prop --advanced` command.

> **Note:**
>
> These advanced properties are set by default and typically are not modified.

An example of an advanced property that you may want to change is the `pool-max-size`. If you have a powerful remote LDAP server and you have configured the proxy so that it receives a maximum of requests, you can increase the `pool-max-size` as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-extension-prop --advanced \
  --extension-name DS-proxy5 \
  --set pool-max-size:2000
```

20.2.1.7 Modifying the LDAP Data Source Monitoring Connection Properties

Using the `dsconfig --advanced` command for the LDAP server extension, you can view or change the following monitoring properties. All properties relate to proactive monitoring unless otherwise specified.

- monitoring-check-interval

  The monitoring check interval is the interval in milliseconds at which the proxy proactive monitoring checks the data source.

  The default value is 30000 milliseconds (30 seconds).

  For more information, see Configuring a Proxy Instance to Monitor Back-End Servers.
- monitoring-connect-timeout

  The maximum time in milliseconds after which the proactive monitoring facility will stop attempting to connect to the remote LDAP server.

  The default value is 5000 milliseconds (5 seconds). 0 means unlimited.
- monitoring-inactivity-timeout

  The time interval in milliseconds after which an idle connection is regularly checked to avoid connection closure by the remote server. The value of this parameter must be superior to the `monitoring-check-interval`.

  The default value is 120000 milliseconds (120 seconds).
- monitoring-ping-timeout

  The maximum time in milliseconds the proactive monitoring attempts to ping the remote server.

  The default value is 5000 milliseconds (5 seconds).
- remote-ldap-server-read-timeout

  The maximum time in milliseconds during which the LDAP Server Extension waits for a response from the remote server before the connection is regarded as having failed. 0 means unlimited.
- remote-ldap-server-connect-timeout

The maximum time in milliseconds during which monitoring attempts to connect to the remote server before the connection is regarded as having failed. 0 means unlimited.

The default is 10000 milliseconds (10 seconds).

20.2.2 Configuring Proxy LDAP Workflow Elements

You can configure the LDAP proxy workflow elements required to communicate with the remote LDAP server by creating a proxy LDAP workflow element.

The following topics describe the LDAP proxy workflow elements configuration:

- Viewing the Existing Proxy LDAP Workflow Elements
- Viewing the Properties of a Proxy LDAP Workflow Element
- Creating a Proxy LDAP Workflow Element
- Modifying the Properties of an LDAP Server Extension

20.2.2.1 Viewing the Existing Proxy LDAP Workflow Elements

Use `dsconfig` command to view a list of all the workflow elements configured on a particular proxy server instance.

To view a list of all the workflow elements configured on a particular proxy server instance, use the `dsconfig list-workflow-elements` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-workflow-elements

Workflow Element : Type               : enabled
-----------------:-------------------:--------
adminRoot        : ldif-local-backend : true
load-bal-we1     : load-balancing     : true
proxy-we1        : proxy-ldap         : true
proxy-we2        : proxy-ldap         : true
```

The proxy workflow elements are the ones with the type `proxy-ldap`.

20.2.2.2 Viewing the Properties of a Proxy LDAP Workflow Element

Use `dsconfig` command to view the proxy workflow element properties.

To view the proxy workflow element properties, use the `dsconfig get-workflow-element-prop` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-workflow-element-prop --element-name proxy-we1

Property                              : Value(s)
--------------------------------------:--------------------
client-cred-mode                      : use-client-identity
enabled                               : true
ldap-server-extension                 : proxy1
remote-ldap-server-bind-dn            : -
remote-ldap-server-bind-password      : -
use-proxy-auth                        : false
```

The following properties are displayed:

- client-cred-mode

  This indicates how the proxy connects to the remote LDAP server. In this example, the status is `use-client-identity`, which means that the proxy will connect to the remote LDAP server with the same credentials that the client used to connect to the proxy.

  This is the default mode.

  For more information, see Configuring Security Between the Proxy and the Data Source.

- enabled

  This indicates if the workflow is enabled (`true`) or not (`false`)

- ldap-server-extension

  This indicates the name of the LDAP server extension with which the workflow element is associated

- remote-ldap-server-bind-dn and remote-ldap-server-bind-password

  This indicates the credentials of the user that the proxy uses to connect to the remote LDAP server when `client-cred-mode` is `use-specific-identity` or `use-proxy-auth`.

20.2.2.3 Creating a Proxy LDAP Workflow Element

You must have configured an LDAP server extension before you create a proxy LDAP workflow element.

To create a proxy LDAP workflow element, use the `dsconfig create-workflow-element` command, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-workflow-element \
  --element-name proxy-we5 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:DS-proxy5
```

The workflow element type must be `proxy-ldap`. The name of the new proxy LDAP workflow element is defined by `element-name`, in this example `proxy-we5`.

The client credential mode (client-cred-mode) indicates how the proxy will connect to the remote LDAP server. In this example, the credential mode is `use-client-identity`, which means that the proxy will connect to the remote LDAP server with the same credentials as those used by the client to connect to the proxy. This is the default mode.

> **Note:**
>
> - If you use Oracle Unified Directory remote LDAP servers and the client credential mode is set to `use-proxy-auth`, the user as which you are connecting *must* exist on the remote LDAP server. If the user does not exist, requests will be rejected. If you cannot guarantee that the user exists on the remote LDAP server, rather set the client credential mode to `use-specific-identity`.
> - If the user deployment performs an internal operations then you must define the root credentials. For example, if you are using RDN changing as described in Performing RDN Changing Configuration, then the root credentials are defined by the following properties:
>
>   `remote-root-dn`
>
>   `remote-root-password`
>
>   These are the credentials for the root user of the remote LDAP server when the server performs internal operations.
> - When managing passwords in a proxy LDAP workflow element (`remote-ldap-server-bind-password` or `remote-root-passord`), the following syntax are valid:
>
>   `<password-value>` or `file://<password-file>`

For more information, see Configuring Security Between the Proxy and the Data Source.

20.2.2.4 Modifying the Properties of a Proxy LDAP Workflow Element

To modify the proxy LDAP workflow element properties, use the `set-workflow-element-prop` command.

You can modify the following properties:

- Set whether the proxy LDAP workflow element is enabled (`true`) or not (`false`)
- Set the client credential mode that is used (`client-cred-mode`)
- Associate an LDAP server extension, to indicate which remote LDAP server to use (`ldap-server-extension`)
- Set the credentials of the user that the proxy uses to connect to the remote LDAP server (`remote-ldap-server-bind-dn` and `remote-ldap-server-bind-password`). The following syntaxes are supported:
  - `<password-value>`
  - `file://<password-file>`

  Passing a password in clear on the command line is supported but not recommended. It is recommended to use a password-file. You can delete the password-file once the command is executed.

For example, if you want to modify the LDAP server extension used by the workflow element in order to use a different remote LDAP server, do the following:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-workflow-element-prop --advanced \
  --element-name proxy-we5 \
  --set remote-ldap-server-bind-dn:uid=Specific\ User,dc=example,dc=com \
```

```
--remote-ldap-server-bind-password:file://pwd-file \
--set ldap-server-extension:DS-proxy3 \
--set client-cred-mode:use-specific-identity
```

20.2.3 Configuring the Bind Mode

You can configure the Bind Mode and its parameters to optimize the server.

The following topics explain how to configure the Bind Mode and its parameters:

- About Configuring the Bind Mode
- Configuring the Bind Mode Parameters to Optimize the Server

20.2.3.1 About Configuring the Bind Mode

Learn how the proxy LDAP workflow element processes an authenticated operation that is executed by the end user.

When an end user executes an authenticated operation, the proxy LDAP workflow element receives the following two distinct operations:

1. A BIND operation that authenticates the user against the remote server.
2. An operation to execute.

When a bind operation is executed, the proxy LDAP workflow element retrieves a connection from the LDAP server extension, performs the BIND operation, then releases the connection.

When the actual operation arrives, the proxy LDAP workflow element again retrieves a connection from the LDAP server extension. If a connection is found that is still bound with the appropriate credentials, that connection is reused. If not, a new connection must be authenticated. This additional authentication operation is called a *silent bind*.

The set of credentials used to perform a silent bind is determined by the *bind mode*, which is a property of the LDAP workflow element. These credentials can be the client credentials or the proxy credentials. Table 20-4 lists the bind modes that are supported by Oracle Unified Directory.

Table 20-4 Supported Bind Modes by Oracle Unified Directory

Mode	Description
`use-client-identity`	Use the client credentials to perform the silent bind.
`use-specific-identity`	Use the proxy credentials to perform the silent bind.

20.2.3.2 Configuring the Bind Mode Parameters to Optimize the Server

You can configure additional parameters to tweak the behavior of the server.

For each of the bind modes described in Table 20-4, you can configure additional parameters to tweak the behavior of the server.

For a description of these parameters, see the following sections:

- About Configuring the `use-client-identity` Bind Mode
- About Configuring the `use-specific-identity` Bind Mode

20.2.3.2.1 About Configuring the `use-client-identity` Bind Mode

If you set the bind mode to `use-client-identity`, then the server uses the client credentials to perform a silent bind — unless specific parameters prevent it from doing so.

For information about the parameters that prevent the server from using the client credentials, see the following sections:

Using Include and Exclude Lists

You can configure the following lists:

- **Include List:** Lists the suffixes that are handled by the remote server.
- **Exclude List:** Lists the suffixes that are not handled by the remote server.

If the client bind DN is a descendant of one DN on the include list, and the client bind DN is not a descendant of any DN on the exclude list, the proxy server uses the client credentials to perform a silent bind. Otherwise the proxy server uses the proxy credentials to perform the silent bind. If both lists are empty, the proxy server always uses the client credentials.

The include and exclude lists are not mutually exclusive and can be used simultaneously. However, it is recommended that you define only one list. In addition, you cannot define the same suffixes in both the lists.

Using the never-bind Parameter

The `never-bind` parameter is applicable whenever the proxy needs to perform a bind with the client credentials. If this flag is set to `true`, the proxy server reads the user entry from the remote data source, and validates the user password itself, instead of forwarding the bind to the remote server.

> **Note:**
>
> The credentials used to read the user entry are proxy credentials, which are defined in the `remote-ldap-server-bind-dn` and `remote-ldap-server-bind-password` properties of the proxy LDAP workflow element.

If the incoming bind operation contains controls that are critical, an error result is returned as controls dedicated to bind operations are incompatible with the `never-bind` feature.

> **Note:**
>
> If the proxy uses its own credentials to read the user entry, then you can add the proxy authorization control to operations to indicate the identity of the client at the origin of the request. The value of the `use-proxy-auth` property determines whether the control should be added.

20.2.3.2.2 About Configuring the `use-specific-identity` Bind Mode

When the bind mode is set to `use-specific-identity`, the proxy server uses the proxy credentials to perform all silent binds.

The proxy credentials are defined in the following properties of the proxy LDAP workflow element: `remote-ldap-server-bind-dn` and `remote-ldap-server-bind-password`.

In `use-specific-identity` bind mode, you can set the following parameters:

Using the use-proxy-auth Parameter

If the `use-proxy-auth` flag is set to `true`, the proxy server adds a proxy authorization control to all requests, except bind requests. The value of the proxy authorization identifier is the client bind DN.

Using the never-bind Parameter

The `never-bind` parameter is applicable whenever the proxy needs to perform a bind with the client credentials. When this flag is set to `true`, the proxy server reads the user entry from the remote data source, and validates the user password itself, instead of forwarding the bind to the remote server.

> **Note:**
>
> The credentials used to read the user entry are proxy credentials, which are defined in the `remote-ldap-server-bind-dn` and `remote-ldap-server-bind-password` properties of the proxy LDAP workflow element.

21

Configuring Load Balancing Using the Proxy

Administration tasks can be performed that are related to load balancing using the proxy `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).
The following topics describe how to perform administration tasks:

- Configuring Load Balancing Using the `dsconfig` Command
- Configuring Load Balancing Using OUDSM

> **Note:**
>
> - To understand more about load balancing, see Overview of Load Balancing Using the Proxy.
> - You can configure load balancing using `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).
>
> For more information about using either option, see Managing the Server Configuration Using `dsconfig`or Accessing Oracle Unified Directory Using OUDSM, respectively.
> - For information about setting up a load balancing deployment during installation, see "To Configure Simple Load Balancing" section in *Installing Oracle Unified Directory*.

21.1 Configuring Load Balancing Using the `dsconfig` Command

The `dsconfig` command is used to configure load balancing, to create a workflow element, and to create an algorithm and routes.

The following topics describe how to configure load balancing using `dsconfig` command:

- Configuring Load Balancing using the dsconfig Command
- Creating a Load Balancing Workflow Element
- Creating a Load Balancing Algorithm
- Creating Load Balancing Routes
- Modifying Load Balancing Properties

21.1.1 Configuring Load Balancing using the dsconfig Command

A load balancing workflow element can only have one load balancing algorithm. However, the same load balancing algorithm is used by all the load balancing routes in the deployment. You can configure load balancing using the `dsconfig` command.

To forward client requests to remote LDAP servers using load balancing, you need the following elements:

- A load balancing workflow element
- A load balancing algorithm
- A load balancing route, for each remote LDAP server

The following examples describe how to configure load balancing using the `dsconfig` command. All of the examples specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN (`-D`), and the bind password file (`-j`), and use the `-X` option to trust all certificates.

1. Create a load balancing workflow element. See Creating a Load Balancing Workflow Element.
2. Create a load balancing algorithm. See Creating a Load Balancing Algorithm.
3. Create one load balancing route for each load balancing workflow element. See Creating Load Balancing Routes.

21.1.2 Creating a Load Balancing Workflow Element

To configure load balancing, you must create a load balancing workflow element using the `dsconfig create-workflow-element` command.

Follow the below given example to create a workflow element:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-workflow-element \
  --element-name load-bal-we1 \
  --type load-balancing \
  --set enabled:true
```

To create a load balancing workflow element, the type must be `load-balancing`. The name of the workflow element is defined by `element-name`, in this example `load-bal-we1`.

21.1.3 Creating a Load Balancing Algorithm

To determine how the requests will be forwarded in a load balancing deployment, you must configure the load balancing algorithm. The load balancing algorithm set determines how client requests will be dispatched across the pool of remote LDAP servers.

The possible load balancing types are: `failover`, `optimal`, `proportional`, or `saturation`.

To create the load balancing algorithm, you must have a load balancing workflow element. See Creating a Load Balancing Workflow Element.

Create a load balancing algorithm using the `dsconfig create-load-balancing-algorithm` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-load-balancing-algorithm \
  --element-name load-bal-we1 \
  --type failover
```

To create a load balancing algorithm, you must indicate the type as `proportional`, `optimal`, `failover`, or `saturation`. The name of the workflow element is defined by `element-name`, in this example `load-bal-we1`.

21.1.4 Creating Load Balancing Routes

You should have one load balancing route per data source. Before you create a load balancing route, the load balancing workflow element and load balancing algorithm must already be created.

To create a load balancing route, use the `dsconfig create-load-balancing-route` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-load-balancing-route \
  --element-name load-bal-we1 \
  --route-name load-bal-route1 \
  --type failover \
  --set workflow-element:proxy-we1 \
  --set add-priority:1 \
  --set bind-priority:2 \
  --set compare-priority:2 \
  --set delete-priority:1 \
  --set extended-priority:2 \
  --set modify-priority:1 \
  --set modifydn-priority:1 \
  --set search-priority:2
```

In this example, `load-bal-route1` is the name of the new load balancing route, `load-bal-we1` is the name of the existing load balancing workflow element, and `proxy-we1` is the name of the LDAP proxy workflow element. The type must be the same as the one defined by the load balancing algorithm associated, in this case `failover`.

The properties set (in this case priority) are related to the type of load balancing created. For more information about the properties of the routes, linked to the algorithm type see Modifying Load Balancing Properties.

21.1.5 Modifying Load Balancing Properties

After a load balancing deployment has been set up, you can modify certain properties, such as the priority, weight, and saturation threshold. Most of these properties are changed at the load balancing route level.

The following sections describe the different settings possible in a load-balancing deployment:

21.1.5.1 Modifying Load Balancing Properties

You can modify the following load balancing properties, depending on the load balancing algorithm:

Failover	Optimal	Proportional	Saturation	Search Filter
add-priority	alert-threshold	add-weight	alert-threshold	priority
bind-priority	saturation-precision*	bind-weight	priority	allowed-attributes
compare-priority	workflow-element	compare-weight	threshold	prohibited-attributes
delete-priority		delete-weight	saturation-precision*	workflow-element
extended-priority		extended-weight	workflow-element	
modify-priority		modify-weight		
modifydn-priority		modifydn-weight		
search-priority		search-weight		
workflow-element		workflow-element		
switch-back flag				

* saturation precision is a property of the LDAP server extension.

To modify load balancing route properties, use the `dsconfig set-load-balancing-route-prop` command.

New routes can be added on a running algorithm, or routes can be deleted or have their priorities modified without the need to restart the server.

> **Note:**
>
> You cannot modify the load balancing algorithm type.
>
> To change a failover load balancing deployment to a proportional one, for example, you must create a new load balancing deployment. See Configuring Load Balancing Using the `dsconfig` Command.

21.1.5.2 Setting the Priority in a Failover Algorithm

In a load balancing deployment that uses the failover algorithm, you can modify the proxy workflow element to change the route that is used, as well as the priority of the route for a given operation type.

In a failover algorithm, a priority of 1 is the highest priority and indicates the main route that will be used for a specific operation type. A route with priority 2 (or more) is the secondary route used in case of failure on the primary route. The priority is set for each operation type. This means that a route with a priority of 1 for Add operations, can have a priority of 2 for Bind and Search operations.

For example, if the route `load-bal-route1` was initially set as the main route with a priority of `1` for Add operations, but you now want to make it the backup route, you can set the priority to `2` using the following command.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-load-balancing-route-prop \
  --element-name load-bal-we1 \
  --route-name load-bal-route1
  --set add-priority: 2
```

> **Note:**
>
> If two routes have the same priority for a given operation type, the choice of the active route which treats the request is random.

21.1.5.3 Setting the `switch-back` Flag

After failover in a load balancing deployment, the backup route continues to handle all incoming requests, even after the priority server that had failed becomes available.

Switch-back or failback to the primary route does not automatically occur unless the `switch-back` flag has been set to `true`. By default, the `switch-back` flag is set to `false`.

The `switch-back` flag is an advanced property. To set the `switch-back` flag to `true`, do the following:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  --advanced set-load-balancing-algorithm-prop \
  --element-name load-bal-we1 \
  --set switch-back:true
```

21.1.5.4 Setting the Saturation Precision for the Optimal or Saturation Algorithm

In a load balancing deployment that uses the optimal or the saturation algorithm, you can set the saturation precision level.

The saturation precision is the delta between two saturation levels, and is used to determine the route with the lowest saturation level. By default, the saturation precision level is set to 5.

If you find that the saturation precision level is too low, and that the routes are changing too frequently, you can modify the saturation precision level as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  --advanced set-extension-prop \
  --extension-name proxy1 \
  --set saturation-precision:10
```

21.1.5.5 Setting the Weight of a Proportional Algorithm

Once you have created a load balancing deployment using the proportional algorithm, you can modify the proxy workflow element to change the route used, as well as the weight of a route.

The weight can be different for each operation type. The value of the weight should be 0 or more, were 0 indicates that the route will not be used for the specified operation.

Using the interactive mode of `dsconfig`, you can see that the following properties can be modified:

```
>>>> Configure the properties of the Proportional Load Balancing Route

        Property          Value(s)
        --------------------------
    1)  add-weight        1
    2)  bind-weight       1
    3)  compare-weight    1
    4)  delete-weight     1
    5)  extended-weight   1
    6)  modify-weight     1
    7)  modifydn-weight   1
    8)  search-weight     1
    9)  workflow-element  proxy-we1
```

For example, if you initially set all your routes to a weight of 1 on all operations, then all the servers will handle an equal ratio of operations. However, if you want a remote LDAP server to handle more search requests than the other servers in the deployment, then you can set its `search-weight` to a higher value, such as 5. To do so, use the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-load-balancing-route-prop \
  --element-name load-bal-we1 \
  --route-name load-bal-route1 \
  --set search-weight:5
```

> **Note:**
>
> To modify the weight for all operations, you must modify the weight for each operation individually.

To modify `load-bal-route1` to handle twice as many operations as the other route, you would set the weight of all operations to `2` (assuming the weight on the other route is set to `1`). In other words, run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-load-balancing-route-prop \
  --element-name load-bal-we1 \
  --route-name load-bal-route1 \
  --set add-weight:2 \
  --set bind-weight:2 \
  --set compare-weight:2 \
  --set delete-weight:2 \
  --set extended-weight:2 \
  --set modify-weight:2 \
  --set modifydn-weight:2 \
  --set search-weight:2
```

If the weight is set to `0` for any operations, the route will not perform the specified operation. For example, if `add-weight` is set to `0`, then `load-bal-route1` will not forward any add requests to the associated remote LDAP server. If all configured routes indicate a weight of `0` for a specific operation, that operation will not be supported.

21.1.5.6 Setting the Threshold for a Saturation Algorithm

Once you have created a load balancing deployment using the saturation algorithm, you can modify the proxy workflow element used, the priority of the route, the saturation threshold, and the saturation threshold alert.

With a saturation algorithm, requests are distributed based on two criteria: the priority of the server and the saturation threshold of the server. The saturation threshold is the limit at which the server is considered “maximized" and service may become degraded. In a load balancing deployment with saturation algorithm, requests are sent to the server with the highest priority (1) until the server reaches the saturation threshold indicated.

For example, if you indicate `load-bal-route1` as the server with the highest priority, with a threshold of 80%, all requests will be sent to `load-bal-route1` until its saturation threshold goes over 80%. Once it exceeds 80%, then requests are routed to the next server in the priority list.

```
>>>> Configure the properties of the Saturation Load Balancing Route

        Property          Value(s)
        --------------------------
    1)  alert-threshold   85
    2)  priority          1
    3)  threshold         80
    4)  workflow-element  proxy-we1
```

To modify the saturation threshold, use the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-load-balancing-route-prop \
  --element-name load-bal-we1 \
  --route-name load-bal-route1 \
  --set threshold:90
```

In this example, the saturation threshold has been set to 90%.

21.1.5.7 Setting the Saturation Threshold Alert

You can use the saturation threshold alert to specify at which point the system administrator will receive a notification indicating that the server has passed the saturation limit.

Generally, the saturation threshold alert is set higher than the saturation limit to indicate if the saturation continues to increase past the saturation threshold (which may indicate a problem). You should set the alert with an acceptable buffer, because there may be a short delay in which saturation continues to increase slightly before requests are forwarded to another route.

To modify the saturation threshold, use the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-load-balancing-route-prop \
  --element-name load-bal-we1 \
  --route-name load-bal-route1 \
  --set alert-threshold:85
```

To perform preventative actions, you can set the saturation threshold alert to a value that is lower than the saturation threshold. (For example, if the main route is a set of load balanced servers, then you could add one or more servers to that set of servers as a preventive action.) This may imply receiving notifications even in cases where the saturation threshold is not reached. That is, a saturation threshold alert is sent, but the saturation limit drops and does not

reach the saturation threshold. However, the requests will only be sent to the next priority route when the saturation threshold is reached.

For more information on setting the notification message, see Configuring Alerts and Account Status Notification Handlers.

21.1.5.8 Setting Client Connection Affinity

When you define a client connection affinity, requests from a specified client connection are routed to the same server, bypassing the specified load balancing algorithm. Client connection affinity is set at the network group level.

To set client connection affinity, use the `dsconfig create-network-group-qos-policy` command. For more information, see Creating a Network Group Quality of Service Policy.

Example of Client Connection Affinity Rejected

When you set the client connection affinity, the load balancing algorithm is bypassed if the defined weight constraints are respected.

For example, assume that the following routes are set with the following weights:

- `LB-route1: add=10, search= 0`
- `LB-route2: add=0, search=10`

It is clear that `LB-route1` receives all the add requests, and `LB-route2` receives all the search requests.

Assume that the load balancing deployment in this example is set with a client connection affinity of `all-requests-after-first-write-request`. If the load balancing deployment receives the following string of requests: Add, Search, Add, typically, the client connection affinity would send the Search request to the same route (`LB-route1`) as the first Add request. However, in this case, since Search requests are not allowed on `LB-route1`, the load balancing algorithm is *not* bypassed by the client affinity.

21.1.5.9 Deleting Load Balancing Elements

This section introduces you to the concept of deleting load balancing elements.

To delete a complete load balancing workflow (including workflow element, algorithm, and routes), you need only delete the load balancing workflow element. When you delete a load balancing workflow element, the associated load balancing algorithm and routes are silently deleted.

21.2 Configuring Load Balancing Using OUDSM

If you have set up a proxy server instance without configuring either load balancing or distribution, you can configure load balancing by using OUDSM.

Before you begin, it is useful to understand the components comprise a load balancing deployment. For more information, see Configuration 1: Simple Load Balancing.

To configure load balancing by using OUDSM:

1. Connect to the proxy server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Home** tab.

3. Under the **Configuration** item, select **Set up Load Balancer**.
4. On the **Load Balancing: Backend Servers** screen, complete the following information:
   - In the **Load Balancing Name** field, provide a name for this load balancing workflow element.
   - Click **Add** to provide the connection details of at least two replicated back end LDAP servers across which client requests will be balanced.

     OUDSM attempts to connect to these back end LDAP servers, to verify that they are accessible. If the connection attempt is unsuccessful, you are prompted to use the server details anyway, or to verify the connection details.
5. When you have added all the back end LDAP servers, click **Next** to continue.
6. On the **Load Balancing: Options** screen, complete the following information:
   - Select the **Load Balancing Algorithm**.
   - Depending on the load balancing algorithm you have selected, specify the relative weight or priority for each back end LDAP server.

   For information about the load balancing algorithms, see Overview of Load Balancing Using the Proxy.
7. When you have specified the load balancing options, click **Next** to continue.
8. On the **Load Balancing: Naming Contexts** screen, click **Add** to specify at least one naming context, or suffix, that will be handled by this proxy instance.
9. When you have added all of the required naming contexts, click **Next** to continue.
10. On the **Load Balancing Setup: Summary** screen, review the load balancing configuration and click **Finish** to complete the configuration.

When you have configured load balancing, you can modify any aspect of the configuration on the OUDSM Configuration tab.

22

Configuring Distribution Using the Proxy

You can choose to configure distribution using the proxy dsconfig or Oracle Unified Directory Services Manager (OUDSM).

- Configuring a Distribution Deployment Using the `dsconfig` Command
- Configuring a Distribution Deployment Using OUDSM

> **Note:**
>
> In some cases, you can choose to configure distribution using `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).
>
> - For information about using the `dsconfig` command, see Managing the Server Configuration Using `dsconfig`.
> - For information about using OUDSM, see Accessing Oracle Unified Directory Using OUDSM.

To forward client requests to remote LDAP servers using distribution, you must configure the following components on the proxy server:

- A distribution workflow element
- A distribution algorithm
- One or more distribution partitions (typically one per remote LDAP server)

A distribution workflow element can only have one distribution algorithm, that defines how data is distributed. A distribution algorithm can use several partitions.

> **Note:**
>
> To understand more about distribution, see Overview of Data Distribution Using the Proxy.

22.1 Configuring a Distribution Deployment Using the `dsconfig` Command

You can configure distribution deployment using the dsconfig command and create a distribution workflow element, algorithm and partitions.

- Configuring Distribution Using dsconfig Command
- Creating a Distribution Workflow Element
- Creating a Distribution Algorithm

- Creating Distribution Partitions
- Managing Modify DN Requests
- Configuring Criticality in Workflows Using `dsconfig`
- Configuring Criticality in Workflow Elements Using `dsconfig`
- Deleting a Distribution Configuration

22.1.1 Configuring Distribution Using dsconfig Command

To configure distribution using the `dsconfig` command, you must set up a distribution deployment.

For information about setting up a distribution deployment during setup, see "To Configure Simple Distribution" section in *Installing Oracle Unified Directory*.

All the commands in the following procedures specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN (`-D`), and the bind password file (`-j`). The examples also use the `-X` option to trust all certificates.

1. Create a distribution workflow element.

   See Creating a Distribution Workflow Element.
2. Create a distribution algorithm.

   See Creating a Distribution Algorithm.
3. Create one partition for each chunk of partitioned data. A partition must be associated with one remote LDAP server, or with a set of replicated remote LDAP servers.
   - For a capacity-based distribution see Creating a `capacity` Distribution Partition.
   - For a lexico or numeric distribution see Creating a `lexico` or `numeric` Distribution Partition.
   - If you are using DN pattern algorithm, see Creating a `dnpattern` Distribution Partition.

22.1.2 Creating a Distribution Workflow Element

To configure distribution, you must create a distribution workflow element using the `dsconfig create-workflow-element` command.

Example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-workflow-element \
  --element-name distrib-we \
  --type distribution \
  --set enabled:true \
  --set base-dn:ou=people,dc=example,dc=com
```

To create a distribution workflow element, the type must be `distribution`. The name of the workflow element is defined by `element-name`, in this example `distrib-we`.

> **Note:**
>
> When declaring the `base-dn` using the `create-workflow-element` subcommand as shown above, ensure that you specify the full tree structure.

To complete the distribution element of your configuration, create the distribution algorithm and the appropriate partitions.

22.1.3 Creating a Distribution Algorithm

To determine how the requests will be forwarded in a distribution deployment, you must configure the distribution algorithm. The algorithm set determines how the data is partitioned and to which partition a request is sent.

The possible distribution types are: `numeric`, `lexico`, or `dnpattern`.

To create the distribution algorithm, you must have a distribution workflow element. See Creating a Distribution Workflow Element.

Create a distribution algorithm using the `dsconfig create-distribution-algorithm` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-distribution-algorithm \
  --element-name distrib-we \
  --type numeric \
  --set distribution-attribute:uid
```

The name of the workflow element is defined by `element-name`, in this example `distrib-we`. The distribution algorithm type must be set as `capacity`, `numeric`, `lexico`, or `dnpattern`. The properties set depend on the algorithm type. In this example, `distribution-attribute` must be set, as the algorithm type is `numeric`.

22.1.4 Creating Distribution Partitions

You can create a distribution partition using the `dsconfig create-distribution-partition` .Before creating a distribution partition, distribution workflow element and distribution algorithm must already be created.

- Creating a `capacity` Distribution Partition
- Creating a `lexico` or `numeric` Distribution Partition
- Creating a `dnpattern` Distribution Partition
- About DN Pattern String Syntax
- Using DN Pattern `negative-match`

22.1.4.1 Creating a `capacity` Distribution Partition

To create a `capacity` distribution partition, the distribution workflow element and distribution algorithm must already be created. You must create one distribution partition per data set.

To create a distribution partition, use the `dsconfig create-distribution-partition` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-partition1 \
  --type capacity \
  --set partition-id:1 \
  --set workflow-element: proxy-we1 \
  --set max-entries:1000
```

> **Note:**
>
> You must create a global index catalog and have the DNs indexed to use the capacity-based algorithm. To create global index catalogs, see Creating a Global Index Catalog Containing Global Indexes.

A distribution partition is identified by both a partition name, in this example, `distrib-partition1` and a partition id. The partition id must be an simple integer, as it will be used for the global index catalog reference. The type must be the same as the one defined by the distribution algorithm associated, in this case `capacity`.

To create a distribution partition, you must also indicate the name of the existing distribution workflow element (`element-name`) that manages the partition (here `distrib-we`), and the name of the next element in the work flow (`workflow-element`), such as an LDAP workflow element (in this example `proxy-we1`). The proxy workflow element indicates the path used to reach the data on the remote LDAP server. For more information on the proxy, see Configuring Communication With Remote LDAP Servers.

When creating a `capacity` distribution partition, you must indicate the maximum number of entries the partition can hold, for example 1000.

22.1.4.2 Creating a `lexico` or `numeric` Distribution Partition

Lexico and numeric distribution are very similar, so you set the same properties when you create a distribution partition for lexico or numeric distribution. You must create one distribution partition per data set.

To create `lexico` or `numeric` distribution partitions, the distribution workflow element and distribution algorithm must already be created.

To create a distribution partition, use the `dsconfig create-distribution-partition` command. For example for a numeric distribution, you might create a partition as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-partition1 \
  --type numeric \
  --set partition-id:1 \
  --set workflow-element: proxy-we1 \
  --set lower-bound:1000 \
  --set upper-bound:2000
```

A distribution partition is identified by both a partition name, in this example, `distrib-partition1` and a partition id. The partition id must be an simple integer, as it will be used for the global index catalog reference. The type must be the same as the one defined by the distribution algorithm associated, in this case `numeric`.

To create a distribution partition, you must also indicate the name of the existing distribution workflow (here `distrib-we`), and the name of the associated workflow element, such as an LDAP workflow element (in this example `proxy-we1`). The proxy workflow element indicates the path used to reach the data on the remote LDAP server. For more information on the proxy, see Configuring Communication With Remote LDAP Servers.

When creating a `lexico` or `numeric` distribution partition, you must indicate the lower and upper boundaries of the partition. The proxy checks to ensure that there is no overlap in the boundaries of any two partitions. This means that you cannot set partition 1 with boundaries 1000-3000 and partition 2 with boundaries 2000-4000.

The upper boundary is exclusive, which means that in the example above, the partitioned data only includes values between 1000 up to 1999. If you want the upper boundary or lower boundary to be unlimited, use the keyword `unlimited`.

The properties set (in this example boundaries) are related to the type of distribution created. For more information about the properties of the partitions, linked to the algorithm type see Configuring a Distribution Deployment Using the `dsconfig` Command.

> **Note:**
>
> For a lexico distribution algorithm, the sort sequence that is used is ASCII.

22.1.4.3 Creating a `dnpattern` Distribution Partition

Before you create a `dnpattern` distribution partition, the distribution workflow element and distribution algorithm must already be created.

To create a `dnpattern` distribution partition, use the `dsconfig create-distribution-partition` command. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-partition5 \
  --type dnpattern \
  --set partition-id:5 \
  --set workflow-element: proxy-we1 \
  --set dn-pattern:uid=[0-9]*[01].*
```

A distribution partition is identified by both a partition name, in this example, `distrib-partition5` and a partition ID. The partition ID is used for the global index catalog reference, and be an simple integer. To create a distribution partition, you must also indicate the name of the existing distribution workflow (here `distrib-we`), and the name of the associated workflow element, such as an LDAP proxy (in this example `proxy-we1`). The type must be the same as the one defined by the distribution algorithm associated, in this case `dnpattern`.

In a distribution scenario that uses a `dnpattern` algorithm, requests are sent to a partition when the request RDNs below the distribution base DN match the DN string pattern. For example, if the distribution base DN is `ou=people,dc=example,dc=com` and the request base DN is `uid=1,ou=people,dc=example,dc=com`, the check against the string pattern is done on the RDN `uid=1`.

Similarly, if the distribution base DN is `ou=people,dc=example,dc=com` and the request base DN is `uid=1,ou=region1,ou=people,dc=example,dc=com`, the check against the string pattern is done on the RDNs `uid=1,ou=region1`.

22.1.4.4 About DN Pattern String Syntax

The DN string pattern must comply with the DN syntax and with a subset of the Java Pattern class.

DN Pattern String	Description
.	any character
\\	backslash
\t	TAB character
[abc]	a, b, or c
[^abc]	any character except a, b, or c
[a-zA-Z]	a through z, or A through Z, inclusive (range)
[a-d[m-p]]	a through d, or m through p (union)
[a-z&&[def]]	d, e, or f (intersection)
[a-z&&[^bc]]	a through z, except for b and c (subtraction)
[A-Z&&[^M-P]]	a through z, and not m through p (subtraction)

The following quantifiers can be used:

DN Pattern Quantifiers	Description
X?	X, once or not at all
X*	X, zero or more times
X+	X, one or more times
X{n}	X, exactly n times
X{n,}	X, at least n times
X{n,m}	X, at least n times but no more than m times

22.1.4.5 Using DN Pattern `negative-match`

The distribution property called `negative-match` allows you to specify the opposite of the DN pattern that should be matched. That is, you specify a DN pattern to be ignored; any value that *does not* match the specified DN pattern will be distributed. By default, the `negative-match` property is set to `false`.

Create a `dnpattern` distribution partition using `negative-match` as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-partition5 \
  --type dnpattern \
  --set partition-id:5 \
  --set workflow-element: proxy-we1 \
  --set dn-pattern:uid=[123]*[0-9].* \
  --set negative-match:true
```

In the example above, since `negative-match` has been set to `true`, any requests for `uid` that does not start with 1, 2, or 3, with n characters following will be forwarded to the partition.

22.1.5 Managing Modify DN Requests

You can modify a DN so that the new entry remains in the same partition as the original entry. By default, the proxy does not allow you to modify the DN to a value that is outside the range of the current partition.

If you want to allow `modifyDN` requests to change the DN to a value that is outside the boundaries of the partition in which the entry is located, set the `force-modify-dn` flag to `true`.

Assume, for example, that you have two partitions: Partition 1 with `uid` boundaries from 0-999 and Partition 2 with `uid` boundaries from 1000-1999. If the `force-modify-dn` flag is set to `true` and you modify the `uid` of an entry from `1` to `1001`, the change will be allowed, but the entry with `uid 1001` will remain in Partition 1. It is not moved to Partition 2.

If you then search for `uid=1001`, the server will return an error, indicating that no such entry is found. To locate the entry, you must use a global index catalog. This ensures that modified entries are always found. To configure a global index catalog, see Configuring Global Indexes Using the Command Line.

To force a modify DN operation, set the `force-modify-dn` flag to `true`, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  --advanced set-workflow-element-prop --element-name distrib-we \
  --set force-modify-dn:true
```

22.1.6 Configuring Criticality in Workflows Using `dsconfig`

The criticality configuration determines the server behavior when a search operation fails. Criticality applies only to search requests. All other requests are processed normally by the server.

You can configure criticality by setting the criticality flag at the workflow level. When a search request is executed on a workflow, then it is executed on several workflows if there are subordinate workflows. The criticality setting of a workflow can be one of the following:

- `true`

  This is the default setting and indicates that the workflow is considered as critical. If a workflow fails to return a result the processing is stopped regardless of whether the execution of the operation was successful on any other workflow.

- `false`

  This setting indicates that the workflow is non-critical. A criticality setting of `false` tells the server that the failure to perform an operation in the workflow is not critical to the overall result. If the non-critical workflow fails to return a result the server simply omits the results (as if the workflow did not return any data), returns a SUCCESS result code to the client, and does not indicate any error.

- `Partial`

  This setting indicates that the workflow is partially critical. This implies that the application can notify its own users that partial results were obtained. If a partially-critical partition fails to return a result because, for example, it is fully saturated or disabled, the server returns an `Admin Limit Exceeded` error. While this is not the expected error, the intention of this setting is to cause client application logic to indicate that not all results are shown.

To set the criticality of a workflow, use the `dsconfig set-workflow-prop` command. For example, the following command sets the criticality of a workflow named `workflow-1` to `true`:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-workflow-prop --workflow-name workflow-1 \
  --set criticality:true
```

22.1.7 Configuring Criticality in Workflow Elements Using `dsconfig`

In a distribution deployment, the *criticality* configuration determines the server behavior when a search operation fails, due to a host error. Criticality applies only to search requests. All other requests are processed normally by the server.

Criticality is configured for each distribution partition in a distribution workflow element. The criticality setting of a distribution partition can be one of the following:

- `true`

  This is the default setting and indicates that the partition is considered as critical. If a partition fails to return a result because, for example, it is fully saturated or disabled, the server returns an UNAVAILABLE error to the client regardless of whether data was found in any other partition.

- `false`

  This setting indicates that the partition is non-critical. A criticality setting of `false` tells the server that the failure to perform an operation in the partition is not critical to the overall result. If the non-critical partition fails to return a result because, for example, it is fully saturated or disabled, the server simply omits the results (as if the partition did not return any data), returns a SUCCESS result code to the client, and does not indicate any error.

- `Partial`

  This setting indicates that the partition is partially critical. This implies that the application can notify its own users that partial results were obtained. If a partially-critical partition fails to return a result because, for example, it is fully saturated or disabled, the server returns an `Admin Limit Exceeded` error. While this is not the expected error, the intention of this setting is to cause client application logic to indicate that not all results are shown.

For all types of workflow element, other than a distribution workflow element, criticality is implicit and is handled as follows:

- Load Balancing: All routes are considered as non critical, because if a route is not functional then it is not taken into consideration by the load balancer while determining the selected route.
- LDAP Proxy Workflow Element: An LDAP server is always considered as critical.
- Local Backend Workflow Element: A local back end server is always considered as critical.

To set the criticality of a distribution partition, use the `dsconfig set-distribution-partition-prop` command. For example, the following command sets the criticality of a partition named `distrib-partition-1` to `true`:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-distribution-partition-prop --element-name distrib-we \
  --partition-name distrib-partition-1 --set criticality:true
```

22.1.8 Deleting a Distribution Configuration

You can delete a distribution configuration by deleting the distribution workflow element.

To delete a complete distribution workflow (including workflow element, algorithm, and partitions), you need only delete the distribution workflow element. When you delete a distribution workflow element, the associated distribution algorithm and partitions are silently deleted.

22.2 Configuring a Distribution Deployment Using OUDSM

You can configure distribution by OUDSM when a proxy server is set up without configuring either load balancing or distribution. Criticality in workflow can be configured by using OUDSM.

The following topics describe how to configure distribution and criticality in workflow using OUDSM:

- Configuring Distribution Using OUDSM
- Configuring Criticality in Workflows Using OUDSM

22.2.1 Configuring Distribution Using OUDSM

If you have set up a proxy server instance without configuring either load balancing or distribution, you can configure distribution by using OUDSM. Before you begin, it is useful to understand the components comprise a distribution deployment.

To configure distribution by using OUDSM:

1. Connect to the proxy server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Home** tab.
3. Under the **Configuration** item, select **Set Up Distributor**.
4. On the **Distribution: Data Partitioning** screen, complete the following information:
   - Select the **Number of Partitions**.
   - Select the **Distribution Algorithm**. For more information about the available distribution algorithms, see Overview of Data Distribution Using the Proxy.
   - Enter the **Naming Context**, or suffix, that will be handled in this distribution deployment.
   - Select the **Network Group** in which the distributor will be configured.
   - Enter the capacity, DN Pattern, or boundaries for each partition, depending on the distribution algorithm that you have selected.
5. When you have entered all of the partition details, click **Next** to continue.
6. On the **Distribution: Server Partitions**, for each partition, click **Add** to enter the connection details of each back-end LDAP server that will hold the partitioned data.

   OUDSM attempts to connect to these back-end LDAP servers, to verify that they are accessible. If the connection attempt is unsuccessful, you are prompted to use the server details anyway, or to verify the connection details.
7. When you have added all of the required servers, click **Next** to continue.

8. On the **Distribution: Global Index** screen, specify the global index details. For more information about global indexes, see Understanding the Global Index Catalog.
9. When you have configured the global index, click **Next** to continue.
10. On the **Distribution: Summary** screen, review the distribution configuration and click **Finish** to complete the configuration.

When you have configured distribution, you can modify any aspect of the configuration on the OUDSM Configuration tab.

For more information, see Configuration 2: Simple Distribution.

22.2.2 Configuring Criticality in Workflows Using OUDSM

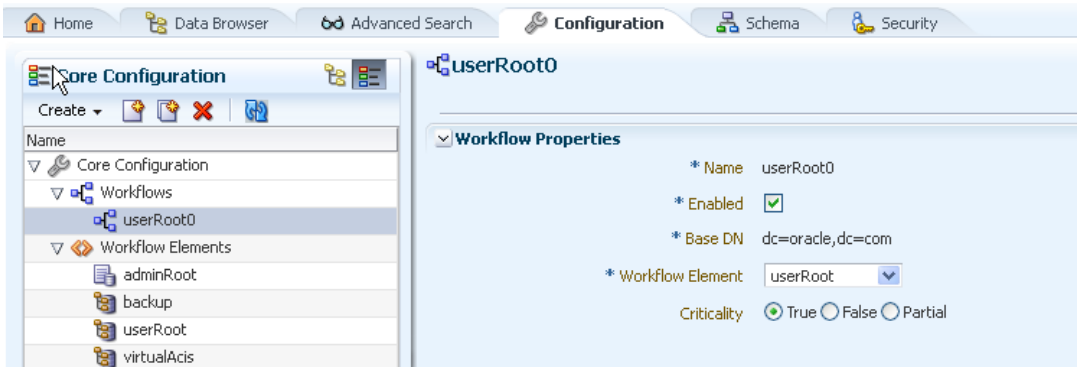
A new parameter known as the criticality flag is added to configure workflows. By default, the criticality flag is set `True`.

The following sections describe how to configure criticality in workflows using OUDSM. For information about configuring criticality using `dsconfig`, Configuring Criticality in Workflows Using `dsconfig`.

To configure criticality in workflows using OUDSM:

1. Connect to the proxy server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.
4. Under **Workflows** element, select the required workflow for which you want to set the criticality flag.
5. Select the criticality value (`True`, `False`, or `Partial`) that you want to set for the workflow. For instance, click **True** to set the criticality for the selected workflow element.

Figure 22-1 Criticality Flag



23
Configuring Integration Using the Proxy

You can configure the server elements that are specific to a proxy instance and integration using `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).

- Retrieving All Attribute Values from an Active Directory Server
- About Integrating with Enterprise User Security Databases
- Updating User Passwords Stored in Active Directory
- Overview of Configuring Pass-Through Authentication
- About Oracle Unified Directory Plug-Ins Configuration
- Configuring a Proxy Instance to Monitor Back-End Servers
- Configuring Global Indexes Using the Command Line
- Configuring Virtual ACIs

> **Note:**
>
> If you configure a load balancing or distribution topology while setting up a proxy instance, then many of these elements are automatically configured.

> **Note:**
>
> In some cases, you can choose to configure integration using `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).
>
> - For information about using the `dsconfig` command, see Managing the Server Configuration Using `dsconfig`.
> - For information about using OUDSM, see Accessing Oracle Unified Directory Using OUDSM.

23.1 Retrieving All Attribute Values from an Active Directory Server

Oracle Unified Directory supports Microsoft Active Directory paging, which enables you to retrieve a complete range of attribute values from the Microsoft Active Directory server.

This section describes how to configure Microsoft Active Directory paging as a workflow element that is relevant only if the leaf of the workflow element chain is connected to an Active Directory server. It also describes how to configure an optional list of attributes to reduce the processing of scanning attributes to detect the range option.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

The topics in this section include:

- Configuring Active Directory Paging Workflow Elements
- Scanning Specific Attributes Returned by an Active Directory

> **Note:**
>
> For more information about Microsoft Active Directory paging, see Understanding How to Retrieve All Attribute Values from an Active Directory Server.

23.1.1 Configuring Active Directory Paging Workflow Elements

Use the following example as a basis for configuring an Active Directory paging workflow element.

This example creates an Active Directory paging workflow element named `ad-paging-we1` that points to the LDAP proxy workflow, `proxy-we1`.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-workflow-element --element-name ad-paging-we1 --type ad-paging \
  --set next-workflow-element:proxy-we1 --set enabled:true
```

23.1.2 Scanning Specific Attributes Returned by an Active Directory

To improve efficiency, you can configure the Active Directory paging workflow element to scan only specific attributes by setting the multi-valued `handled-attributes` property of the workflow element. You can add as many values for this property as required.

By default all attributes are scanned. This can have a direct impact on performance. To reduce the performance impact, list only the attributes that need to be scanned as values of the `handled-attributes` property.

The following example modifies the workflow element created in the previous example to scan only for the `memberOf` attribute:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-workflow-element-prop --element-name ad-paging-we1 \
  --set handled-attributes:memberOf
```

23.2 About Integrating with Enterprise User Security Databases

You can integrate Oracle Unified Directory and Enterprise User Security to leverage user identities stored in an LDAP-compliant directory service without any additional synchronization.

When integrated with Enterprise User Security, Oracle Unified Directory supports the following:

- Microsoft Active Directory

- Novell eDirectory
- Oracle Unified Directory
- Oracle Directory Server Enterprise Edition

For more information about Oracle Enterprise User Security, see Introducing Enterprise User Security in the *Oracle Database Enterprise User Security Administrator's Guide*. For detailed instructions on configuring Oracle Unified Directory and Enterprise User Security to work together, see Integrating Oracle Unified Directory with Oracle Enterprise User Security.

23.3 Updating User Passwords Stored in Active Directory

The Ad Password workflow element enables Oracle Unified Directory LDAP client applications to update user passwords stored in Microsoft Active Directory and Active Directory Lightweight Directory Services (AD LDS) using the LDAP protocol.

For an overview of the Ad Password workflow element, see Overview of Enabling LDAP Clients to Update User Passwords Stored in Active Directory.

The following topics describe how to configure an Ad Password workflow element and its required components:

- Setting Up an Oracle Unified Directory Proxy Server
- Creating and Configuring an Ad Password Workflow Element
- Creating a Workflow for the Ad Password Workflow Element
- Adding the Workflow to a Network Group

> **Note:**
>
> The examples in this section uses the `dsconfig` command-line utility to create and configure the Ad Password workflow element and its required components. The descriptions of these examples mention key options and properties you must set.
>
> For the description of all `dsconfig` subcommands and options, see Oracle Unified Directory Command-Line Interface Reference.

23.3.1 Setting Up an Oracle Unified Directory Proxy Server

The Ad Password workflow element requires an Oracle Unified Directory proxy server as the interface between LDAP clients and the Active Directory or AD LDS server.

The example in this section applies to both use cases described in Creating and Configuring an Ad Password Workflow Element .

To setup a proxy server instance using command-line tools on a UNIX or Linux system:

1. Ensure that your `JAVA_HOME` environment variable is set to a supported JVM installation.
2. Run the `oud-proxy-setup` script to set up the proxy server instance:

```
$ export INSTANCE_NAME=ad-oud-proxy-instance
$ OUD_HOME/oud-proxy-setup --cli -p oud-port --adminConnectorPort admin-port
-D "cn=Directory Manager" -j password-file
```

In this example:

- `ad-oud-proxy-instance` is the proxy instance directory name. This example sets the `INSTANCE_NAME` environment variable to this directory before running the `oud-proxy-setup` script.
- *`oud-port`* is the LDAP port used to access the proxy server instance.
- *`admin-port`* is the administration port.
- *`password-file`* contains the administrator password.

On Windows systems, run the `oud-proxy-setup.bat` script.

For more information, see Setting up Oracle Unified Directory as a Proxy Server in *Installing Oracle Unified Directory*.

23.3.2 Creating and Configuring an Ad Password Workflow Element

You can create and configure Ad password workflow element when SSL is Required for Only Password Modifications and for all LDAP operations.

The following topics explain how to create and configure Ad password workflow element:

- Creating and Configuring an Ad Password Workflow Element
- Configuring an Ad Password Workflow Element When SSL is Required for Only Password Modifications
- Configuring an Ad Password Workflow Element When SSL is Required for All LDAP Operations

23.3.2.1 Creating and Configuring an Ad Password Workflow Element

When you create and configure an Ad Password workflow element and its supporting components, you have two choices.

The two choices are:

- **SSL is Required for Only Password Modifications**

  For this use case, you must define the Ad Password workflow element with both the `secure-proxy-workflow-element` and `next-workflow-element` properties. The `secure-proxy-workflow-element` must use an LDAP server extension configured with `remote-ldap-server-ssl-policy` set to `always`.

  In this use case, operations to modify a password will be routed to the `secure-proxy-workflow-element` and will take place over SSL. Operations not related to password modifications will be routed to the `next-workflow-element` and will take place over a non-SSL connection.

  See Configuring an Ad Password Workflow Element When SSL is Required for Only Password Modifications.

- **SSL is Required for All LDAP Operations**

  For this use case, your proxy LDAP workflow element must point to an LDAP server extension that always uses SSL (`remote-ldap-server-ssl-policy` set to `always`). You can define the Ad Password workflow element with only a `next-workflow-element` property. All operations will then be routed to the `next-workflow-element` and will take place over SSL.

  See Configuring an Ad Password Workflow Element When SSL is Required for All LDAP Operations.

23.3.2.2 Configuring an Ad Password Workflow Element When SSL is Required for Only Password Modifications

The following tasks create and configure an Ad Password workflow element and its required components when an SSL connection to the Active Directory or AD LDS server is required only for LDAP operations that perform password modifications. Other LDAP operations are performed over a non-SSL connection.

These tasks include:

- Creating the LDAP Server Extensions
- Creating the Proxy LDAP Workflow Elements
- Creating an Ad Password Workflow Element

23.3.2.2.1 Creating the LDAP Server Extensions

This use case requires two LDAP server extensions to communicate with the remote Active Directory or AD LDS server:

- An LDAP server extension for LDAP operations that do not require an SSL connection.

  To create an LDAP server extension that does not require an SSL connection from the LDAP clients to the Active Directory or AD LDS server:

  ```
  $ dsconfig create-extension \
    --set enabled:true \
    --set remote-ldap-server-address:adserver.example.com \
    --set remote-ldap-server-port:389 \
    --set remote-ldap-server-ssl-port:636 \
    --type ldap-server \
    --extension-name adserver \
    --hostname localhost \
    --port 4444 \
    --trustAll \
    --bindDN cn=directory\ manager \
    --bindPasswordFile pwd.txt \
    --no-prompt
  ```

  In this example:

  – `remote-ldap-server-ssl-policy` is not set in this command, so the default value of `never` specifies a non-SSL connection.

  – `extension-name` is set to `adserver` for a non-SSL connection.

  – `enabled` must be set to `true` to enable the LDAP server extension for use in the server.

  and

- An LDAP server extension for LDAP operations that require an SSL connection.

  To create an LDAP server extension that requires an SSL connection from the LDAP clients to the Active Directory or AD LDS server:

  ```
  $ dsconfig create-extension \
    --set enabled:true \
    --set remote-ldap-server-address:adserver.example.com \
    --set remote-ldap-server-ssl-policy:always \
    --set remote-ldap-server-port:389 \
    --set remote-ldap-server-ssl-port:636 \
  ```

```
  --set ssl-trust-all:true \
  --type ldap-server \
  --extension-name adsecureserver \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `remote-ldap-server-ssl-policy` is set to `always`, so an SSL connection is always used to access the Active Directory or AD LDS server
- `extension-name` is set to `adsecureserver` to indicate an SSL connection.
- `enabled` must be set to `true` to enable the LDAP server extension for use in the server.

23.3.2.2.2 Creating the Proxy LDAP Workflow Elements

This use case requires two proxy LDAP workflow elements to communicate with the remote Active Directory or AD LDS server:

- A proxy LDAP workflow element for LDAP operations that do not require an SSL connection to the Active Directory or AD LDS server.

  To create a proxy LDAP workflow element that does not require an SSL connection from the LDAP clients to the Active Directory or AD LDS server:

```
$ dsconfig create-workflow-element \
  --set client-cred-mode:use-client-identity \
  --set enabled:true \
  --set ldap-server-extension:adserver \
  --type proxy-ldap \
  --element-name adproxy \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

  In this example:

  - `client-cred-mode` is set to the `use-client-identity` bind mode, which specifies that the proxy will connect to the Active Directory or AD LDS server with the same credentials used by the client to connect to the proxy.
  - `element-name` specifies the name of the proxy LDAP workflow element as `adproxy`.
  - `ldap-server-extension` specifies the name of the LDAP server extension as `adserver`.
  - `enabled` must be set to `true` to enable the Ad Password workflow element for use in the server.

  and

- A secure proxy LDAP workflow element for LDAP operations that require an SSL connection to the Active Directory or AD LDS server.

  To create a secure proxy LDAP workflow element that uses an SSL connection from the LDAP clients to the Active Directory or AD LDS server:

```
$ dsconfig create-workflow-element \
  --set client-cred-mode:use-client-identity \
  --set enabled:true \
  --set ldap-server-extension:adsecureserver \
  --type proxy-ldap \
  --element-name adsecureproxy \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `client-cred-mode` is set to the `use-client-identity` bind mode, which specifies that the proxy will connect to the Active Directory or AD LDS server with the same credentials used by the client to connect to the proxy.
- `element-name` specifies the name of the secure proxy LDAP workflow element as `adsecureproxy`.
- `ldap-server-extension` specifies the name of the LDAP server extension as `adsecureserver`.
- `enabled` must be set to `true` to enable the proxy LDAP workflow element for use in the server.

23.3.2.2.3 Creating an Ad Password Workflow Element

This use case requires an Ad Password workflow element that can handle LDAP operations that support both SSL and non-SSL connections to the Active Directory or AD LDS server.

To create this Ad Password workflow element:

```
$ dsconfig create-workflow-element \
  --set enabled:true \
  --set next-workflow-element:adproxy \
  --set secure-proxy-workflow-element:adsecureproxy \
  --type ad-password \
  --element-name ADPasswordWFE \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `type` must be `ad-password`.
- `element-name` specifies the workflow name as `ADPasswordWFE`.
- `next-workflow-element` routes LDAP operations to the proxy LDAP workflow element named `adproxy`, which routes operations over a non-SSL connection.
- `secure-proxy-workflow-element` routes LDAP operations to the proxy LDAP workflow element named `adsecureproxy`, which then routes operations over an SSL connection.
- `enabled` must be set to `true` to enable the Ad Password workflow element for use in the server.

23.3.2.3 Configuring an Ad Password Workflow Element When SSL is Required for All LDAP Operations

The following configuration tasks create and configure the components for an Ad Password workflow element when all LDAP operations between the LDAP clients and Active Directory or AD LDS server must be performed over an SSL connection:

- Creating an LDAP Server Extension
- Creating a Proxy LDAP Workflow Element
- Creating an Ad Password Workflow Element

23.3.2.3.1 Creating an LDAP Server Extension

The Ad Password workflow element requires an LDAP server extension to communicate with the remote Active Directory or AD LDS server.

To create an LDAP server extension that always uses an SSL connection:

```
$ dsconfig create-extension \
  --set enabled:true \
  --set remote-ldap-server-address:adserver.example.com \
  --set remote-ldap-server-port:389 \
  --set remote-ldap-server-ssl-port:636 \
  --set remote-ldap-server-ssl-policy:always \
  --set ssl-trust-all:true \
  --type ldap-server \
  --extension-name adsecureserver \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `type` must be `ldap-server`.
- `extension-name` defines the name of the new extension as `adsecureserver`.
- `remote-ldap-server-ssl-policy` property is set to `always`, so that all connections made from the proxy to the remote Active Directory or AD LDS server will use SSL, regardless of how clients connect to the proxy server.
- `enabled` must be set to `true` to enable the LDAP server extension for use in the server.

23.3.2.3.2 Creating a Proxy LDAP Workflow Element

This use case requires a secure proxy LDAP workflow element to communicate with the remote Active Directory or AD LDS server over SSL.

To create a secure proxy LDAP workflow element:

```
$ dsconfig create-workflow-element \
  --set client-cred-mode:use-client-identity \
  --set enabled:true \
  --set ldap-server-extension:adsecureserver \
  --type proxy-ldap \
  --element-name adsecureproxy \
```

```
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `type` must be `proxy-ldap`.
- `element-name` specifies the name of the new proxy LDAP workflow element as `adsecureproxy`.
- `ldap-server-extension` is set to `adsecureserver`, which is the name of the LDAP server extension with the `remote-ldap-server-ssl-policy` property set to `always`.
- `client-cred-mode` is set to `use-client-identity`, which specifies that the proxy will connect to the Active Directory or AD LDS server with the same credentials used by the client to connect to the proxy.
- `enabled` must be set to `true` to enable the proxy LDAP workflow element for use in the server.

23.3.2.3.3 Creating an Ad Password Workflow Element

This use case requires an Ad Password workflow element that can handle LDAP operations that always require an SSL connection to the Active Directory or AD LDS server.

This Ad Password workflow element requires only the `next-workflow-element` property. All operations will take place over an SSL connection.

To create this Ad Password workflow element:

```
$ dsconfig create-workflow-element \
  --set enabled:true \
  --set next-workflow-element:adsecureproxy \
  --type ad-password \
  --element-name ADPasswordWFE \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `next-workflow-element` property is set to the secure proxy LDAP workflow element named `adsecureproxy`.
- `type` must be `ad-password`.
- `element-name` is set to `ADPasswordWFE`.
- `enabled` must be set to `true` to enable the Ad Password workflow element for use in the server.

23.3.3 Creating a Workflow for the Ad Password Workflow Element

The Ad Password workflow element must be associated with a workflow.

The following example applies to both use cases described in Creating and Configuring an Ad Password Workflow Element .

To create a workflow for the Ad Password workflow element:

```
$ dsconfig create-workflow \
  --set base-dn:dc=example,dc=com \
  --set enabled:true \
  --set workflow-element:ADPasswordWFE \
  --type generic \
  --workflow-name adworkflow \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

In this example:

- `workflow-element` is set to the Ad Password workflow element named `ADPasswordWFE`.
- `workflow-name` is set to `adworkflow`.
- `enabled` must be set to `true` to enable the Ad Password workflow element for use in the server.

23.3.4 Adding the Workflow to a Network Group

Network groups are the single entry point of client requests to Oracle Unified Directory. A workflow must be registered with at least one network group, although it can be attached to several network groups.

You must add the workflow from the previous task to either an existing network group or a new network group.

The following example applies to both use cases described in Creating and Configuring an Ad Password Workflow Element . It adds the `adworkflow` workflow to the default network group (`network-group`).

```
$ dsconfig set-network-group-prop \
  --group-name network-group \
  --add workflow:adworkflow \
  --hostname localhost \
  --port 4444 \
  --trustAll \
  --bindDN cn=directory\ manager \
  --bindPasswordFile pwd.txt \
  --no-prompt
```

23.4 Overview of Configuring Pass-Through Authentication

You can implement pass-through authentication by using `dsconfig` command and OUDSM.

The following topics describe how to implement pass-through authentication:

- Configuring Pass-Through Authentication
- Prerequisites for Configuring Pass-Through Authentication
- Best Practices for Configuring Pass-Through Authentication
- Configuring Pass-Through Authentication Using `dsconfig`
- Understanding Pass-Through Authentication Configuration Using OUDSM

23.4.1 Configuring Pass-Through Authentication

To configure Pass-Through Authentication, you need to know prerequisites and best practices information.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

> **Note:**
>
> For more information about pass-through authentication, see Understanding Pass-Through Authentication.

To implement pass-through authentication:

1. Review the following prerequisites and best practices information and then, if necessary, perform the prerequisite steps.
   - Prerequisites for Configuring Pass-Through Authentication
   - Best Practices for Configuring Pass-Through Authentication.
2. Configure pass-through authentication.
   - If you are using `dsconfig`, see Configuring Pass-Through Authentication Using `dsconfig`.
   - If you are using OUDSM, see Configuring Pass-Through Authentication Using `dsconfig`.
3. Create a workflow using the pass-through authentication workflow element or the Kerberos workflow element.
   - If your user entries are stored on a remote LDAP server, then see Configuring Pass-Through Authentication for Different Servers.
   - If your user entries are stored on a Kerberos server, then see Configuring Pass-Through Authentication for a Kerberos Server.
4. Insert the workflow created in Step 3 into an existing network group or a new network group.

23.4.2 Prerequisites for Configuring Pass-Through Authentication

Ensure these prerequisites before you start configuring Pass-Through Authentication.

Before attempting to implement pass-through authentication, read Understanding Pass-Through Authentication. In addition, you must keep the following steps in mind while configuring pass-through authentication:

1. Create or identify a workflow element for the Auth provider.
   - If the credentials are stored on a remote LDAP server, then you must create a LDAP server extension and a Proxy LDAP workflow element for this remote server, and then use this Proxy LDAP workflow element as `auth-provider-workflow-element` as described in Configuring Pass-Through Authentication for Different Servers.

     Keep in mind that you must configure `remote-root-dn` and `remote-root-password` parameters, and set the `client-cred-mode=use-client-identity` bind mode.

     For more information about how to create an LDAP server extension, see Configuring LDAP Server Extensions.
   - If the credentials are stored inside a Kerberos server, then you must create a Kerberos workflow element, and then use this Kerberos workflow element as `auth-provider-workflow-element` in step 1 as described in Configuring Pass-Through Authentication for a Kerberos Server.
2. Create or identify a workflow element for the User provider.
   - If the user entries are stored on a remote LDAP server, then you must create a LDAP server extension and a Proxy LDAP workflow element for this remote server, and then use this Proxy LDAP workflow element as `user-provider-workflow-element` as described in Configuring Pass-Through Authentication for Different Servers.

     Keep in mind that you must configure `remote-root-dn` and `remote-root-password` parameters.

     For more information about how to create an LDAP server extension, see Configuring LDAP Server Extensions.
   - If the user entries are stored locally, then you must create a Local Backend workflow element, and then use this Local Backend workflow element as `user-provider-workflow-element` as described in Configuring Pass-Through Authentication for Different Servers.

23.4.3 Best Practices for Configuring Pass-Through Authentication

Oracle Unified Directory recommends the following best practices to configure pass-through authentication:

- If you are using different suffixes for user-provider workflow element and authentication-provider workflow element, then it is recommended to define virtual ACIs to protect your data. Your virtual ACIs are defined using `pta-suffix`.
- If the authentication provider is a Kerberos workflow element, then you should not specify any join rule or authentication suffix.
- If the authentication provider is a Proxy workflow element, then you are required to configure a `remote-root-dn`.

- If the user provider is a Proxy workflow element, then you are required to configure a `remote-root-dn`. You must configure the proxy server carefully, because it performs silent bind.

23.4.4 Configuring Pass-Through Authentication Using `dsconfig`

You can configure pass-through authentication using the `dsconfig` command for different servers and a Kerberos server.

The following examples configure pass-through authentication with a remote LDAP server that stores the credentials with a base DN `dc=auth,dc=com`. The Oracle Unified Directory instance stores the user entries locally below the `dc=user,dc=com` suffix.

Here the remote LDAP server acts an Authentication Server, whereas Oracle Unified Directory acts as the User Server.

All the commands in the following procedures specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN (`-D`), and the bind password file (`-j`). The examples use the `-X` option to trust all certificates.

This section contains the following topics:

- Configuring Pass-Through Authentication for Different Servers
- Configuring Pass-Through Authentication for a Kerberos Server

23.4.4.1 Configuring Pass-Through Authentication for Different Servers

You can configure Pass-Through Authentication for different servers, as described in the below sections:

To configure pass-through authentication for a remote LDAP server:

1. Create a LDAP server extension for the LDAP server that stores the credentials.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" \
-j pwd-file -X -n create-extension --extension-name authServer \
--type ldap-server --set enabled:true \
--set remote-ldap-server-address:authHostname \
--set remote-ldap-server-port:1389 \
```

   For more information about how to create an LDAP server extension, see Creating an LDAP Server Extension.

2. Create a Proxy LDAP workflow element using the LDAP server extension that you have created in Step 1.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --element-name authProxy --type proxy-ldap \
--set enabled:true --set client-cred-mode:use-client-identity \
--set ldap-server-extension:authServer \
--set remote-root-dn:cn=administrator,cn=users,dc=auth,dc=com \
--set remote-root-password:********
```

   For more information about how to create a Proxy LDAP workflow element, see Creating the Proxy LDAP Workflow Elements.

3. Create a Local Backend workflow element to store the user entries locally, below the `dc=user,dc=com` suffix.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --element-name userWfe --type db-local-backend \
--set enabled:true --set base-dn:"dc=user,dc=com"
```

4. Create the Pass-Through Authentication workflow element.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --element-name ptaWfe \
--type pass-through-authentication --set enabled:true \
--set auth-provider-workflow-element:authProxy \
--set user-provider-workflow-element:userWfe \
--set pta-auth-suffix:dc=auth,dc=com --set pta-suffix:dc=user,dc=com \
--set pta-user-suffix:dc=user,dc=com
```

23.4.4.2 Configuring Pass-Through Authentication for a Kerberos Server

You can configure Pass-Through Authentication for a Kerberos server as described in the below section.

To configure pass-through authentication with kerberos workflow element:

1. Create a Kerberos Auth Provider workflow element.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --type kerberos-auth-provider \
--element-name kerberosWfe --set enabled:true
```

2. Create a Local Backend workflow element to store the user entries locally, below the `dc=user,dc=com` suffix.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --element-name userWfe --type db-local-backend \
--set enabled:true --set base-dn:dc=user,dc=com
```

3. Create a Pass-Through Authentication workflow element.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element --element-name ptaWfe \
--set auth-provider-workflow-element:kerberosWfe --set enabled:true \
--set user-provider-workflow-element:userWfe --type pass-through-authentication
```

23.4.5 Understanding Pass-Through Authentication Configuration Using OUDSM

You need to create a work flow element to configure a pass-through authentication.

For information about configuring a pass-through authentication workflow element using OUDSM, see Creating a Workflow Element.

23.5 About Oracle Unified Directory Plug-Ins Configuration

You can use the Oracle Unified Directory plug-in API to extend existing directory server functionality when you have a particular requirement and Oracle Unified Directory does not provide the necessary functionality to accommodate that requirement.

For example, you might configure a plug-in to customize LDAP operations or programmatically manipulate results.

> **Note:**
>
> - For detailed information about developing and deploying Oracle Unified Directory plug-ins, see Building and Deploying an OUD Plug-In in
>
> *Oracle Fusion Middleware Developing Plug-Ins for Oracle Unified Directory.*.
> - For more information about Oracle Unified Directory plug-ins, see Understanding Oracle Unified Directory Plug-Ins.

23.6 Configuring a Proxy Instance to Monitor Back-End Servers

The proxy server periodically performs health check to determine the status and in turn the availability of the host. You can configure the time interval between checks using the `monitoring-check-interval` property of `ldap-server-extension` instance.

The `monitoring-check-interval` property of proxy configuration against all back-end servers, defines the time internal (in milliseconds) between health checks scheduled by the proxy server.

You can set the `monitoring-check-interval` property using the `dsconfig` command as follows:

```
dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --no-prompt\
set-extension-prop \
--extension-name proxy1 \
--set monitoring-check-interval:50000 \
```

23.7 Configuring Global Indexes Using the Command Line

Global indexes map entries to a specific distribution partition to speed up search and modify operations in distributed topologies. A global index maps entries based on a unique attribute, such as a phone number. Lists of global indexes are contained in a global index catalog. A proxy instance can contain one or more global index catalogs.

> **Note:**
>
> To configure and manage global indexes and global index catalogs, you must enable specific controls on the remote servers, particularly the LDAP Pre-Read Control and the CSN Control. For more information, see LDAP Controls and Operations Reference.

This section contains the following topics:

- Configuring Global Index Catalogs Using `gicadm`
- Replicating Global Index Catalogs
- Configuring Controls Required by the Global Index Catalog with Oracle Unified Directory

23.7.1 Configuring Global Index Catalogs Using `gicadm`

Global index catalogs are stored in a Berkeley database under `INSTANCE_DIR/OUD/catalogs`. To ensure high availability of a distributed topology, replication of global index catalogs is recommended.

.For more information, see Replicating Global Index Catalogs.

The `gicadm` command is located in the server instance directory:

- For UNIX: `INSTANCE_DIR/OUD/bin/gicadm`
- For Windows: `INSTANCE_DIR\OUD\bat\gicadm.bat`

For more information, see gicadm.

The procedures in this section assume that the proxy is deployed in a distribution architecture and presume that you are using the default proxy administration port (4444). This section contains the following topics:

- Creating a Global Index Catalog Containing Global Indexes
- Viewing Global Index Catalog Properties
- About Modifying the Global Index Catalog Properties
- Modifying the Global Index Catalog Properties
- Modifying Multi-Valued Global Index Catalog Properties
- Resetting Global Index Catalog Properties to the Default Values
- Viewing Global Index Properties
- Importing Content into a Global Index Catalog
- Exporting Contents of a Global Index Catalog to a Directory
- Associating a Global Index Catalog With a Distribution Element
- Disassociating a Global Index Catalog From a Distribution Element
- Adding a Global Index to a Global Index Catalog
- Removing a Global Index From a Global Index Catalog

23.7.1.1 Creating a Global Index Catalog Containing Global Indexes

To create global indexes, you must first create global index catalogs, as described in the following procedure. This procedure describes how to create global index catalogs, create and add global indexes, and add data to the global indexes. You can add the data to your global indexes later, if you prefer.

Before you begin, the proxy must be deployed for distribution.

1. Use the `gicadm` command to create a global index catalog:

   ```
   $ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
   create-catalog --catalogName sampleCatalog
   ```

   The catalog name must be unique.

2. Create and add at least one global index to the global index catalog.

The following command creates a global index of `telephoneNumber` attribute values and adds that global index to the global index catalog that was created in the previous step.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  add-index --catalogName sampleCatalog --attributeName telephoneNumber
```

You can use the `add-index` subcommand later to add additional global indexes to the global index catalog.

3. Associate the global index catalog to a distribution.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  associate --catalogName sampleCatalog \
  --distributionWorkflowElement myDistributionName
```

For information about workflow elements, see Configuring Workflow Elements Using `dsconfig`. For information about distribution, see Configuring a Distribution Deployment Using the `dsconfig` Command.

4. Use the `split-ldif` command to generate multiple files from one LDIF file.

The `split-ldif` command separates the content of one LDIF file into several LDIF files based on the distribution algorithm configured with your proxy. It can also generate files that contain data to load in a global index. You should use `split-ldif` during global index initialization if the remote LDAP servers will contain data that must be indexed when you start your Directory service. If you plan to start without data in your directory, you can skip this step.

For information on the `split-ldif` command, including examples on how to use the command to populate a global index with one or several indexed attributes, see split-ldif.

5. Use the `gicadm import` command to import data into the global index.

For more information, see Importing Content into a Global Index Catalog.

23.7.1.2 Viewing Global Index Catalog Properties

Global index catalog properties are related to global index catalog **replication**. To view a list of the global index catalog properties and an explanation of their use, see About Modifying the Global Index Catalog Properties.

To view all the properties of a global index catalog, use the `gicadm` command with the `get-catalog-prop` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  get-catalog-prop --catalogName sampleCatalog --property all
```

The output will be similar to the following.

```
Property   : Value(s)
-------------------:-------------------------------
replication-server : localhost:3390, localhost:4390
server-id          : 4247
window-size        : 100
heartbeat-interval : 1000
group-id           : 1
```

To view the value for a specific global index catalog property, specify the property.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  get-catalog-prop --catalogName sampleCatalog --property heartbeat-interval
```

23.7.1.3 About Modifying the Global Index Catalog Properties

Global index properties are related to the replication of global index catalogs. The following global index catalog properties are available:

- `replication-server`: Lists the servers in the replication topology, in the format *host:port*. Do *not* use the `set-catalog-prop` command to modify this property. Instead, use the `enable-replication` command.
- `server-id`: Specifies a unique identifier for the global index within the global index catalog replication domain. Each instance within the same global index catalog replication domain must have a different server ID. An instance which is a member of multiple global index catalog replication domains may use the same server ID for each of its global index catalog replication domain configurations.

  Syntax: 1 <= INTEGER <= 65535 or text. This property should not be modified.
- `window-size`: Specifies the window size that the instance will use when communicating with replication servers. Default value is 100.

  Syntax: 0 <= INTEGER or text.
- `heartbeat-interval`: Specifies the heartbeat interval that the instance will use when communicating with replication servers. The instance expects a regular heartbeat from the replication server within the specified interval. If a heartbeat is not received within this interval, the instance closes its connection and connects to another replication server.

  Syntax: 100 ms <= DURATION (ms)
- `group-id`: The id associated with a specific replicated domain. This value defines the group id of the replicated domain. The replication system will preferably connect and send updates to replicate to a replication server with the same group id as itself.

  Syntax: 1 <= INTEGER <= 127

> **Note:**
>
> This property should not be modified.

23.7.1.4 Modifying the Global Index Catalog Properties

You can modify the global index catalog properties using `gicadm` command.

For a list of the global index catalog properties, see About Modifying the Global Index Catalog Properties.

Use the `gicadm` command with the `set-catalog-prop` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  set-catalog-prop --catalogName sampleCatalog --set property:value
```

For example, one of the properties that can be modified is the heartbeat interval. In this case, use:

```
--set heartbeat-interval:500
```

23.7.1.5 Modifying Multi-Valued Global Index Catalog Properties

For multi-valued global index or global index catalog properties, you can add or remove a value using the `--add` or `--remove` options.

For global index catalog, only the property `replication-server` can be multi-valued. For multi-valued global index properties, use the `set-index-prop` subcommand instead

1. To add a value, use the `gicadm` command with the `set-catalog-prop` subcommand.

   ```
   $ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
     set-catalog-prop --catalogName sampleCatalog --add replication-server:hostname
   ```

2. To remove a value from a multi-valued property, use the `gicadm` command with the `set-catalog-prop` subcommand.

   ```
   $ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
     set-catalog-prop --catalogName sampleCatalog \
     --remove replication-server:hostname
   ```

23.7.1.6 Resetting Global Index Catalog Properties to the Default Values

If you have modified any of the global index catalog properties and want to reset them to the default values, use the following procedure.

Use the `gicadm` command with the `set-catalog-prop` subcommand.

For example, to reset the heartbeat interval:

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  set-catalog-prop --catalogName sampleCatalog --reset heartbeat-interval
```

23.7.1.7 Viewing Global Index Properties

To view the properties of a global index, use the `gicadm` command with the `get-index-prop` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  get-index-prop --catalogName sampleCatalog --attributeName telephoneNumber \
  --property all
```

The properties should be similar to the following:

```
Property Names                                : Property Values
----------------------------------------------:---------------------------------
global-index-deleted-entry-retention-timeout : 500
db-cleaner-min-utilization                    : 50
db-log-file-max                               : 10000000
db-checkpointer-bytes-interval                : 20000000
db-checkpointer-wakeup-interval               : 30
db-num-lock-tables                            : -
db-num-cleaner-threads                        : -
db-txn-no-sync                                : false
db-txn-write-no-sync                          : true
je-property                                   : -
db-directory                                  : catalogs
db-directory-permissions                      : 700
global-index-catalogs-shared-cache            : global-index-catalogs-shared-cac
global-index-attribute                        : telephoneNumber
```

> **Note:**
>
> Generally, these values should not be modified.

23.7.1.8 Importing Content into a Global Index Catalog

You can import the contents of a specific file into one or multiple global indexes in a global index catalog. You must specify the name of the catalog into which the content of the file is to be imported. You can filter the content of the file to data related to a particular index by optionally providing the `attributeName` parameter.

The data file to be imported can be created by executing the `split-ldif` command or from executing the `gicadm export` command, for example.

To import the contents of a file into a global index catalog, use the `gicadm` command with the `import` subcommand. For example:

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  import --file /usr/local/import-file --catalogName sampleCatalog
```

If the proxy server stops while a `gicadm import` task is being executed, the global index catalog workflow element is disabled. In this case, re-enable the global index catalog workflow element by using `dsconfig`, as follows, where *sampleCatalog* is the name of the global index catalog:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-workflow-element-prop --element-name sampleCatalog set enabled:true
```

23.7.1.9 Exporting Contents of a Global Index Catalog to a Directory

To export the contents of a global index catalog to a directory, use the `gicadm` command with the `export` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  export --exportDirectory directory-path --catalogName sampleCatalog
```

23.7.1.10 Associating a Global Index Catalog With a Distribution Element

To associate a global index catalog with a distribution element, use the `gicadm` command with the `associate` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  associate --catalogName sampleCatalog --distributionWorkflowElement element-name
```

When the global index catalog is associated with a distribution workflow element, the global index catalog will be listed in the properties of the distribution. To confirm which global index catalog is associated to a distribution, use the `dsconfig get-workflow-element-prop` command. For information on workflow elements, see Configuring Workflow Elements Using `dsconfig`.

23.7.1.11 Disassociating a Global Index Catalog From a Distribution Element

To disassociate a global index catalog from a distribution topology, you must know the distribution workflow element with which the global index catalog is associated. To confirm the name of the distribution workflow element that is using the global index catalog, view the properties of the distribution topology by using the `dsconfig get-workflow-element-prop` command.

To disassociate a global index catalog from a distribution workflow element, use the `gicadm` command with the `disassociate` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
 disassociate --distributionWorkflowElement element-Name
```

23.7.1.12 Adding a Global Index to a Global Index Catalog

To add a new global index to an existing global index catalog, for example to map a new attribute, use the following procedure. This procedure creates and adds the global index to the global index catalog. It is not possible to create a global index without adding it to a global index catalog.

Before you begin, you must already have configured a global index catalog.

Use the `gicadm` command with the `add-index` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  add-index --catalogName sampleCatalog --attributeName telephoneNumber
```

23.7.1.13 Removing a Global Index From a Global Index Catalog

To remove a global index from a global index catalog, Use the `gicadm` command with the `remove-index` subcommand.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  remove-index --catalogName sampleCatalog --attributeName telephoneNumber
```

23.7.2 Replicating Global Index Catalogs

To ensure high availability, global index catalogs should be replicated.

You can use a standard hardware load balancer and configure global index catalogs replication in a deployment, as shown by the graphic in Configuration 7: Multiple Replicated Proxies.

This section contains the following topics:

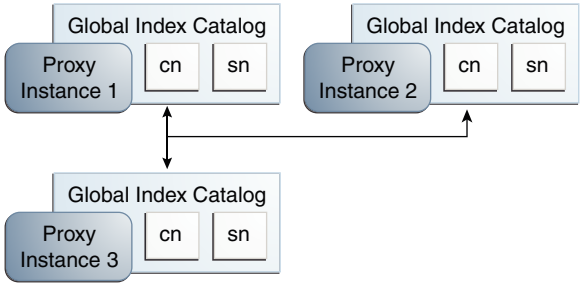
- Creating a Replicated Topology and Enable Global Index Catalog Replication
- Enabling Global Index Catalog Replication
- Initializing Global Index Catalog Replication
- Disabling Global Index Catalog Replication
- Viewing the Status of a Replicated Global Index Catalog Configuration
- Logging of Replication Activities
- Lifecycle Examples for Replicated Global Index Catalogs

23.7.2.1 Creating a Replicated Topology and Enable Global Index Catalog Replication

This section describes how to create a replicated topology with three proxy instances and enable global index catalog replication.

This section describes how to create a replicated topology with three proxy instances and enable global index catalog replication, as illustrated in Figure 23-1.

Figure 23-1 Replicated Global Index Catalogs



To create a replicated topology and enable global index catalog replication:

1. Install at least two proxy instances in your server topology.

   These instances should be on separate physical machines, for redundancy.

2. Configure a global index catalog for each instance of the proxy in your topology and add one or more global indexes.

   For more information on configuring a global index catalog using the `gicadm` command, see Creating a Global Index Catalog Containing Global Indexes.

3. Enable global index catalog replication.

   The proxy instance whose global index catalog is to be replicated across the topology is referred to, for the purposes of CLI syntax, as the *local* instance, while the other proxy instance declared in the command is referred to as the *remote* instance. For more information on running the `gicadm enable-replication` command, see Enabling Global Index Catalog Replication.

   Repeat this step for each proxy that is part of your replicated topology.

4. Choose a proxy instance on which to initialize replication. Consider which proxy instance has the most up to date global index catalog content.

   Otherwise, you can import the LDIF file to each proxy that is part of the topology. See Importing Content into a Global Index Catalog.

5. On the proxy instance chosen in the previous step, run the `gicadm initialize-replication --all` command. For more information, see Initializing Global Index Catalog Replication.

> **Note:**
>
> When using a global index catalog with replicated remote LDAP servers, only one remote LDAP server must handle write operations if such operations can concurrently modify the same value *and* if that value is indexed. For this, you could set the weights in your load balancing workflow element to direct all write traffic to the same server. For more information, see Modifying Load Balancing Properties.

23.7.2.2 Enabling Global Index Catalog Replication

This command configures replication but does not initialize replication. The command is executed on the local host, declared by the `-h` option, using the administration port of the local host. The remote host is declared by the `--remoteHost` option, and must be a fully qualified host name or IP address. The command creates a global index catalog replication administrator with a bind ID of *adminUID*.

If you created global index catalogs during installation, the global index administrator is already created, with the same password as the directory manager. For more information on installing a distribution deployment with global index, see "To Configure Simple Distribution" section in *Installing Oracle Unified Directory*.

To enable replication of global index catalogs, use the `gicadm enable-replication` command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  enable-replication --catalogName sampleCatalog --adminUID adminUID
  --localReplicationPort 8989 --remoteReplicationPort 8989 \
  --remoteAdminPort 4444 --remoteHost host
```

This command updates the proxy configuration to replicate the content of the global index catalog called *sampleCatalog* on the local host. If one of the proxy instances in the topology already replicates the global index catalog, this command updates the configuration of all other proxy instances in the topology. It is therefore sufficient to execute the `gicadm enable-replication` once for the first two proxy instances in the topology, and once for each new proxy instance that is added to extend the topology.

The proxy instance on which you execute the command must be the instance whose replication port is declared by the `--localReplicationPort` option. It is this local instance whose global index catalog is replicated across the topology later by the `gicadm initialize-replication` command. The `--remoteReplicationPort` option will replicate the content of the global index catalog called *sampleCatalog* from the local instance on to the remote instance. The `--remoteAdminPort` is the administration port of the remote proxy instance.

You can declare the password for the local proxy instance in a file, by using the `--adminPasswordFile` suboption.

You can optionally declare a DN for binding to the remote server by using the `--remoteBindDN` suboption and the password for the remote proxy instance in a file, by using the `--remoteBindPasswordFile` suboption. If you do not declare these, the global administrator that is declared by `--adminUID` will be used to bind.

You can also optionally require the communication through the replication port of the local server to be secure, using the `--localSecureReplication` suboption, and the communication through the replication port of the remote server to be secure, using the `--remoteSecureReplication` suboption.

23.7.2.3 Initializing Global Index Catalog Replication

This command initializes the content of the global index catalog called *sampleCatalog* from the proxy instance on the server declared by the `-h` option to all instances that are part of the topology. The port specified is the administration port, and not the replication port.

1. To initialize the replication of a global index catalog to all proxy instances that are part of the replication topology, use the `gicadm initialize-replication --all` as follows:

   ```
   $ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
     initialize-replication --catalogName sampleCatalog \
     --adminUID adminUID --all
   ```

2. Check that replication is complete by using the `gicadm status-replication` command.

   If replication is complete, the status for all proxy instances in the topology is given as `running replicated`.

   Replication must be complete before restarting any proxy instances in the topology, for example after applying a patch.

   For information about using the `gicadm status-replication` command, see Viewing the Status of a Replicated Global Index Catalog Configuration.

23.7.2.4 Disabling Global Index Catalog Replication

To disable replication of global index catalogs, use the `gicadm disable-replication` command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  disable-replication --catalogName sampleCatalog --adminUID adminUID
```

The `gicadm disable-replication` command must be executed for each proxy instance in the topology on which you want to disable replication.

23.7.2.5 Viewing the Status of a Replicated Global Index Catalog Configuration

To view basic configuration information about a replicated global index catalog, use the `gicadm status-replication` command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
  status-replication --catalogName sampleCatalog --adminUID adminUID
```

If you do not declare a catalog name, status information for all replicated global index catalogs is displayed.

23.7.2.6 Logging of Replication Activities

Replication logs are stored in the replication repair logs. Changes are recorded in the change logs.

For information on accessing these logs, see Accessing Logs.

When replicating global index catalogs, provision disk space for change logs. By default, these logs store changes for a 24 hour period. Approximately 100Mb is required for 300,000 write operations. With the default value of 24 hours, the log must be configured based on the expected size of the service during that period. A hint is to provision approximately 150Gb for 5

000 modifications per second over 24 hours. For information how to configure logs, see Configuring Logs.

23.7.2.7 Lifecycle Examples for Replicated Global Index Catalogs

This section describes several typical lifecycle examples in which events take place in a replication topology.

The basic replication topology used in all of these examples is the one created in Creating a Replicated Topology and Enable Global Index Catalog Replication.

The following topics explain, with examples, how to restart a global index catalogs, add a global index to the global index catalog topology, overwrite the contents of global index catalogs and add a proxy to the replicated topology:

- Restarting a Global Index Catalog
- Adding a Global Index to a Replicated Global Index Catalog Topology
- Overwriting the Contents of Replicated Global Index Catalogs
- Adding a Proxy to a Replicated Topology

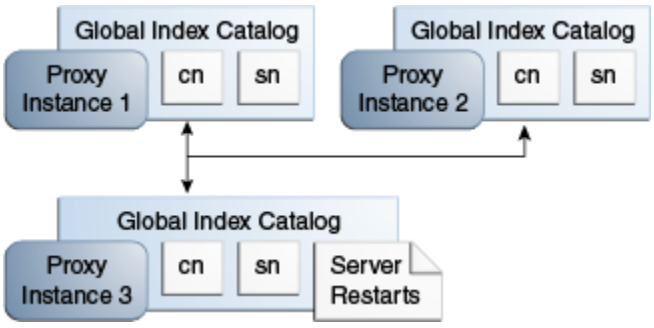
23.7.2.7.1 Restarting a Global Index Catalog

This section describes the procedure to restart a global index catalog.

The example illustrated by Figure 23-2, shows three proxy instances running with a replicated global index catalog. If proxy instance 3 goes down or is stopped, for whatever reason, follow these steps to ensure that the three instances of the proxy are replicated.

1. Issue the `start-ds` command on proxy instance 3.
2. You can verify if replication is complete by executing the `gicadm status-replication` command, as described in Viewing the Status of a Replicated Global Index Catalog Configuration.

Figure 23-2 Restarting a Global Index Catalog



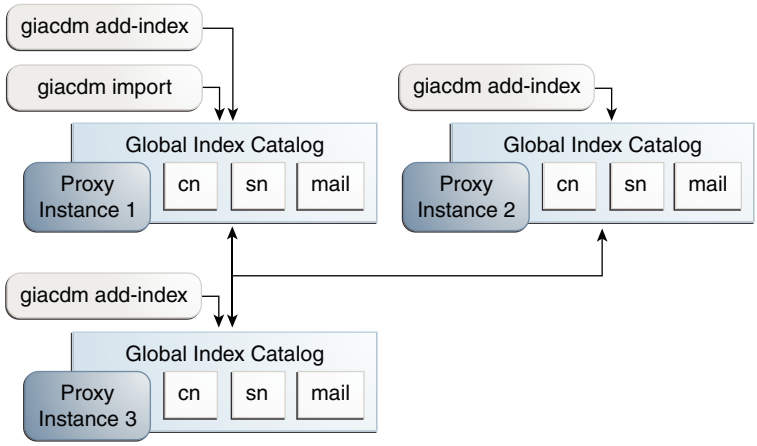
23.7.2.7.2 Adding a Global Index to a Replicated Global Index Catalog Topology

This section describes the procedure to add a global index to a replicated global index catalog topology.

The example illustrated by Figure 23-3, shows three proxy instances running with a replicated global index catalog. If you want to add an attribute, for example, `mail`, to the replicated global index catalog, follow these steps.

1. First, run the command `gicadm add-index mail` on each of the three proxy instances.
2. Export the directory data under the distribution route from one of the remote LDAP servers to an LDIF file named *file1* by using `export-ldif`.
3. Run `split-ldif` to generate GIC content in the specified directory.
4. On proxy instance 1, execute the command `gicadm import --importDirectory directory-name`.
5. On proxy instance 1, execute the `gicadm initialize-replication --all` command. This command pushes the changes from proxy 1 to all the other proxies in the topology, and adds the new global index.

Figure 23-3 Adding a Global Index to a Replicated Global Index Catalog Topology



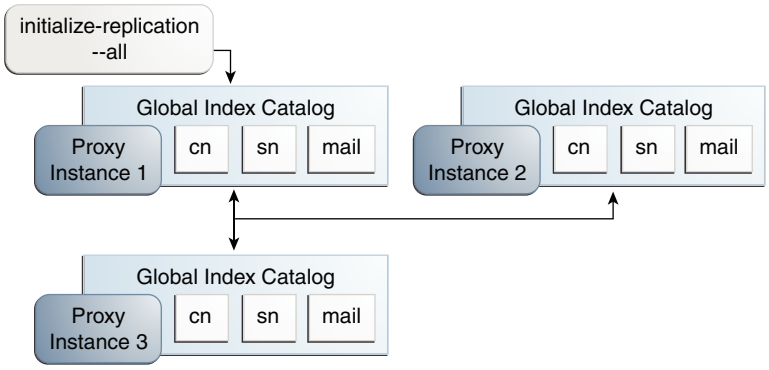
23.7.2.7.3 Overwriting the Contents of Replicated Global Index Catalogs

This section describes the procedure to overwrite the contents of replicated global index catalogs.

The example illustrated by Figure 23-4, shows three proxy instances running with a replicated global index catalog. To overwrite the content of the global index catalogs on proxy instances 2 and 3 with the content of the global index catalog on instance 1, follow these steps.

- On proxy instance 1, execute the `gicadm initialize-replication --all` command. This replaces the content of the global index catalog on proxy instance 2 and 3 with the content of the global index catalog on proxy instance 1.

Figure 23-4 Overwriting the Contents of Replicated Global Index Catalogs



23.7.2.7.4 Adding a Proxy to a Replicated Topology

This section describes the procedure to add a proxy to a replicated topology.

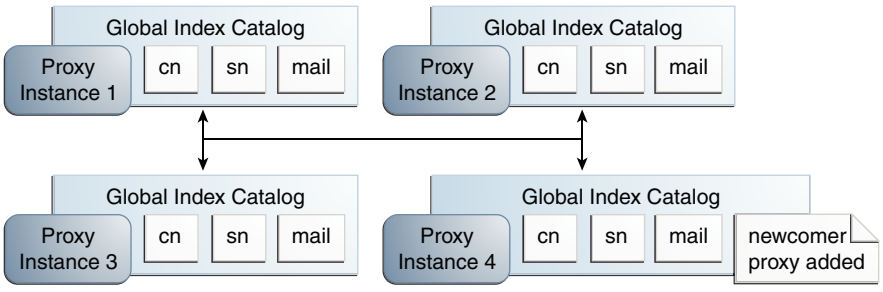
The example illustrated by Figure 23-5, shows three proxy instances running with a replicated global index catalog. To add a fourth proxy instance with a replicated global index catalog, follow these steps on the new proxy instance.

1. On the new proxy instance 4, execute the `gicadm create-catalog` command.
2. Run the commands`gicadm add-index cn`, `gicadm add-index sn`, and `gicadm add-index mail`.
3. Execute the `gicadm associate` command.
4. Run the following command:

   ```
   gicadm enable-replication --localReplicationPort replication port of
   instance 4 --remoteHost name or IP address of host running instance 1
   ```

   This command configures replication between instance 1 and instance 4.
5. Run the `initialize replication --from proxy 1` command.

Figure 23-5 Adding a Proxy to a Replicated Topology



23.7.3 Configuring Controls Required by the Global Index Catalog with Oracle Unified Directory

If you are using the proxy server with an Oracle Unified Directory directory server as the LDAP data source, the connections between the proxy and directory servers must be bound using

the directory server's administrator ID. Otherwise, some configuration is required on the directory server to allow the global index catalog to function correctly.

If the global ACIs for controls have not been modified, then use the `ldapmodify` command to apply the following changes to the directory server:

```
dn: cn=Access Control Handler,cn=config
changetype: modify
add: ds-cfg-global-aci
ds-cfg-global-aci:
(targetcontrol="2.16.840.1.113730.3.4.2 || 2.16.840.1.113730.3.4.17 |
| 2.16.840.1.113730.3.4.19 || 1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 |
| 2.16.840.1.113730.3.4.16 || 1.3.6.1.1.13.1 || 1.3.6.1.4.1.42.2.27.9.5.9")
(version 3.0; acl "Anonymous control access"; allow(read) userdn="ldap:///anyone";)
 ds-cfg-global-aci: (targetattr="createTimestamp||creatorsName||modifiersName|
|modifyTimestamp||entryDN||entryUUID||subschemaSubentry||aclRights||aclRightsInfo")
(version 3.0; acl "User-Visible Operational Attributes"; allow (read,search,compare)
 userdn="ldap:///anyone";)
```

If you are deleting the ACI from an Oracle Unified Directory 11*g* R1 directory instance, then you must delete the following ACI:

```
dn: cn=Access Control Handler,cn=config
changetype: modify
delete: ds-cfg-global-aci
ds-cfg-global-aci: (targetcontrol="2.16.840.1.113730.3.4.2 ||
 2.16.840.1.113730.3.4.17 || 2.16.840.1.113730.3.4.19 ||
 1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 ||
 2.16.840.1.113730.3.4.16") (version 3.0; acl "Anonymous control access";
 allow(read) userdn="ldap:///anyone";)
ds-cfg-global-aci: (targetattr="createTimestamp||creatorsName||modifiersName||
modifyTimestamp||entryDN||entryUUID||subschemaSubentry")
(version 3.0; acl "User-Visible Operational Attributes"; allow
 (read,search,compare) userdn="ldap:///anyone";)
```

If you are deleting the ACI from an Oracle Unified Directory 11*g* R2 directory instance, then you must delete the following ACI:

```
dn: cn=Access Control Handler,cn=config
changetype: modify
delete: ds-cfg-global-aci
ds-cfg-global-aci: (targetcontrol="2.16.840.1.113730.3.4.2 ||
 2.16.840.1.113730.3.4.17 || 2.16.840.1.113730.3.4.19 ||
 1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 ||
 2.16.840.1.113730.3.4.16 || 2.16.840.1.113894.1.8.31")
(version 3.0; acl "Anonymous control access"; allow(read)
 userdn="ldap:///anyone";)
```

> **Note:**
>
> The preceding OIDs are correct for an unmodified configuration of Oracle Unified Directory. If you change the default OIDs, modify the command include the correct OIDs.

The following controls are required for global index catalogs:

- The Pre-Read Control, with OID = `1.3.6.1.1.13.1`
- The CSN Control, with OID = `1.3.6.1.4.1.42.2.27.9.5.9`

23.8 Configuring Virtual ACIs

Each workflow is associated to an access control group which defines the list of ACIs that apply to operations handled by this workflow.

By default, an access control group is created known as "Local Backends." This access control group contains all ACIs coming from user data. You cannot delete it. If virtual ACIs are disabled for a workflow, then you must specify Local Backends as the access control group for that workflow. For the workflow for which virtual ACIs are enabled, you can specify any access control group.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

The following topics describe how to configure virtual ACIs for a workflow using `dsconfig` command and OUDSM:

- Configuring Virtual ACIs Using `dsconfig`
- Configuring Access Control Groups Using OUDSM

23.8.1 Configuring Virtual ACIs Using `dsconfig`

You can enable and disable virtual ACIs for a workflow and configure a replication for virtual ACIs using `dsconfig`.

The following topics describe how to enable and disable virtual ACIs for a workflow:

- Enabling Virtual ACIs for a Workflow
- Disabling Virtual ACIs for a Workflow
- Configuring Replication for Virtual ACIs

23.8.1.1 Enabling Virtual ACIs for a Workflow

To enable virtual ACIs for a specific workflow, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
set-workflow-prop --workflow-name workflow1 --set virtual-aci-mode:true \
--set access-control-group:group1
```

In this example, `group1` references an access control group. This access control group can be either `Local Backends`, which is created by default or any other access control group that you have created. For more information about creating access control groups, see Configuring Access Control Groups With `dsconfig`.

23.8.1.2 Disabling Virtual ACIs for a Workflow

To disable virtual ACIs for a specific workflow, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
set-workflow-prop --workflow-name workflow1 --set virtual-aci-mode:false \
--set access-control-group:"Local Backends"
```

> **Note:**
>
> You must bear the following points in mind when you disable virtual ACIs for a workflow:
>
> - If you disable virtual ACIs, you must specify "Local Backends" as the access control group for this workflow.
> - Disabling virtual ACIs for a specific workflow does not delete virtual ACIs from the associated access control group.

23.8.1.3 Configuring Replication for Virtual ACIs

You can configure replication of virtual ACIs through the `--advanced` mode of the `dsreplication` command.

To configure replication of virtual ACIs:

1. Enable replication of virtual ACI.

```
$ dsreplication enable \
  --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPasswordFile1 pwd-file1 --replicationPort1 8989 \
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 pwd-file2--replicationPort2 8989 \
  --adminUID admin --adminPasswordFile admin-pwd-file \
--advanced --baseDN virtual-acis -X -n
```

2. Initialize replication.

```
$ dsreplication initialize --advanced --baseDN virtual-acis \
  --adminUID admin --adminPasswordFile admin-pwd-file \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 -X -n
```

23.8.2 Configuring Access Control Groups Using OUDSM

You can create access control elements for Oracle Unified Directory Proxy Servers using OUDSM.

Perform the following steps to create access control elements:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **General Configuration** element.

   The properties are displayed in the right hand pane.

4. Expand **Access Control Groups**.
5. Click **Add** to specify at least one local back end, that will be handled by this proxy instance.

   Click **Delete**, if you want to delete those access control groups that are not associated with any workflow. Deleting an access control group will delete all ACIs contained in that access control group.

24

Configuring Virtualization

You can configure a virtual directory view of repositories and optimize search results from the virtual directory
The following topics describe how to configure a virtual directory view of repositories:

- Configuring a Virtual Directory View of Your Repositories
- Optimizing Search Results From a Virtual Directory
- Adding the `memberof` User Attribute to `person` Entries
- Performing DN Renaming
- Performing RDN Changing Configuration
- Configuring Transformations
- Configuring SAML XASP
- Deploying ForkJoin Workflow Element Configuration Model
- Configuring DynamicGroup Workflow Element

This chapter also gives an overview of DN Renaming, RDN Changing and Transformations Configurations.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid Oracle Directory Service Plus license.

> **Note:**
>
> You can choose to configure some virtualization elements using `dsconfig` or Oracle Unified Directory Services Manager (OUDSM).
>
> - For information about using the `dsconfig` command, see Managing the Server Configuration Using `dsconfig`.
> - For information about using OUDSM, see Accessing Oracle Unified Directory Using OUDSM.

24.1 Configuring a Virtual Directory View of Your Repositories

You can create and configure a Join workflow element to create a virtual directory view of your repositories by using `dsconfig` command or OUDSM.

The following topics describe configuring a virtual directory view:

- Prerequisites for Creating the Join Workflow Element

- Creating a Join Workflow Element Using the `dsconfig` Command
- Creating a Join Workflow Element Using OUDSM

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid Oracle Directory Service Plus license.

24.1.1 Prerequisites for Creating the Join Workflow Element

Before creating the Join workflow element, you must configure the participating workflow elements so you can link to them from the Join workflow element configuration.

For example, consider a scenario with two separate Proxy LDAP workflow elements:

- The first Proxy LDAP workflow element, `we-proxy1`, will be linked to the primary participant of the Join workflow element configuration.
- The second Proxy LDAP workflow element, `we-proxy2`, will be linked to the secondary participant of the Join workflow element configuration.

> **Note:**
>
> For more information about creating Proxy LDAP workflow elements, see Configuring Proxy LDAP Workflow Elements.

Assume there is an entry in the `we-proxy1` data source as follows:

```
dn:cn=john,cn=users,dc=com1
objectclass:inetorgperson
cn:john
sn:doe
uid:jdoe
title:PMTS
description: This entry is from we-proxy1
```

Next, assume there is an entry in the `we-proxy2` data source as follows:

```
dn: sn=doe,cn=employees,dc=com2
empid: jdoe
cn:John
sn:doe
department: Sales
manager: userid=smith,cn=users,dc=com2
description: This entry is from we-proxy2
objectclass:inetorgperson
```

The joined-entry returned from Join Workflow element would be:

```
dn:cn=john,cn=users,dc=join
objectclass:inetorgperson
cn:john
sn:doe
uid:jdoe
```

```
empid: jdoe
title:PMTS
description: This entry is from we-proxy1
description: This entry is from we-proxy2
manager: userid=smith,cn=users,dc=join
department: Sales
```

24.1.2 Creating a Join Workflow Element Using the `dsconfig` Command

You can create and configure a Join workflow element topology, based on the scenario using the two Proxy LDAP workflow elements.

The two Proxy LDAP workflow elements are described in Prerequisites for Creating the Join Workflow Element.

> **Note:**
>
> The following steps assume that you have already created the participating workflow elements.

To configure a Join workflow element topology:

1. Create a Join workflow element, named `we-join`.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X
-n create-workflow-element --set enabled:true --set join-suffix:dc=join
--type join --element-name we-join

>>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server hostname or IP address [ip]:

Directory server administration port number [4444]:

Administrator user bind DN [cn=Directory Manager]:

Password for user 'cn=Directory Manager':


>>>> Configure the properties of the Join Workflow Element

        Property               Value(s)
        ---------------------------------------------------------------
    1)  dn-attribute           manager, member, memberof, uniquemember
    2)  enabled                true
    3)  join-suffix            dc=join
    4)  populate-joinedentrydn  false

    ?)  help
    f)  finish - create the new Join Workflow Element
    q)  quit

Enter choice [f]: f

The Join Workflow Element was created successfully
```

2. Create a primary participant, named `jp-p1`, that is linked to the Proxy LDAP workflow element named, `we-proxy1`.

```
dsconfig create-join-participant --element-name we-join \
--set participant-dn:dc=com1 \
--set participating-workflow-element:we-proxy1 \
--set primary-participant:true --type generic --participant-name jp-p1 \
```

Provide the following information to create a primary participant:

```
>>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server hostname or IP address [ip]:

Directory server administration port number [4444]:

Administrator user bind DN [cn=Directory Manager]:

Password for user 'cn=Directory Manager':


>>>> Configure the properties of the Join Participant

         Property                        Value(s)
         ------------------------------------------------------------------------
    1)   enabled-operation               compare, delete, modify, search
    2)   join-condition                 By default, no join condition is
                                         defined. That is all entries
                                         satisfying the original search filter
                                         are considered for join.
    3)   joiner-type                    one-to-one
    4)   non-retrievable-attribute       By default, the non-retrievable list
                                         is empty, which means that all
                                         attributes are retrievable.
    5)   non-storable-attribute          By default, the non-storable list is
                                         empty, which means that all attributes
                                         are storable.
    6)   participant-bind-priority       0
    7)   participant-criticality         true
    8)   participant-dn                  dc=com1
    9)   participants-join-rule          ""
    10)  participating-workflow-element  we-proxy1
    11)  primary-participant             true
    12)  retrievable-attribute           By default, the retrievable list is
                                         empty, which means that all attributes
                                         are retrievable.
    13)  storable-attribute              By default, the storable list is
                                         empty, which means that all attributes
                                         are storable.

    ?)   help
    f)   finish - create the new Join Participant
    q)   quit

Enter choice [f]: f

The Join Participant was created successfully.
```

3. Create a secondary participant, named `jp-p2`, that is linked to the Proxy LDAP workflow element named, `we-proxy2`.

```
dsconfig create-join-participant --element-name we-join \
--set participant-dn:dc=com2 \
--set participating-workflow-element:we-proxy2 \
--set primary-participant:false --type generic --participant-name jp-p2 \
--set participants-join-rule:jp-p1.uid=jp-p2.empid
```

Provide the following information to create a secondary participant:

```
>>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server hostname or IP address [ip]:

Directory server administration port number [4444]:

Administrator user bind DN [cn=Directory Manager]:

Password for user 'cn=Directory Manager':


>>>> Configure the properties of the Join Participant

         Property                        Value(s)
         ---------------------------------------------------------------------
    1)   enabled-operation               compare, delete, modify, search
    2)   join-condition                 By default, no join condition is
                                        defined. That is all entries
                                        satisfying the original search filter
                                        are considered for join.
    3)   joiner-type                    one-to-one
    4)   non-retrievable-attribute      By default, the non-retrievable list
                                        is empty, which means that all
                                        attributes are retrievable.
    5)   non-storable-attribute         By default, the non-storable list is
                                        empty, which means that all attributes
                                        are storable.
    6)   participant-bind-priority      0
    7)   participant-criticality        true
    8)   participant-dn                 dc=com2
    9)   participants-join-rule         jp-p1.uid=jp-p2.empid
    10)  participating-workflow-element we-proxy2
    11)  primary-participant            false
    12)  retrievable-attribute          By default, the retrievable list is
                                        empty, which means that all attributes
                                        are retrievable.
    13)  storable-attribute             By default, the storable list is
                                        empty, which means that all attributes
                                        are storable.

    ?)   help
    f)   finish - create the new Join Participant
    q)   quit

Enter choice [f]: f

The Join Participant was created successfully.
```

4. To specify which Join policy type to use for a Join workflow element, configure the `ds-cfg-join-policy` parameter. For example, `--set join-policy:left-outer-join`.

24.1.3 Creating a Join Workflow Element Using OUDSM

You can create a Join workflow element using the OUDSM graphical user interface.

> **Note:**
>
> For information, see Creating a Workflow Element.

24.2 Optimizing Search Results From a Virtual Directory

To help you more efficiently view or retrieve data from virtual data sources, Oracle Unified Directory provides two workflow elements that automatically narrow search results.

You can insert the `GetRidOfDuplicate` or `HideByFilter` workflow elements into any workflow chain that returns search results.

This section includes the following topics:

- Eliminating Duplicate Entries from Search Results Using the `GetRidofDuplicate` Workflow Element
- Filtering Search Results Using the `HideByFilter` Workflow Element

For more information about Oracle Unified Directory workflows, see “OUD Plug-Ins and Workflows" in *Oracle Fusion Middleware Developer's Guide for Oracle Unified Directory*.

24.2.1 Eliminating Duplicate Entries from Search Results Using the `GetRidofDuplicate` Workflow Element

The `GetRidofDuplicate` workflow element removes, from search results for the current search operation, all the entries whose DN has already been returned to the client application. This is useful when a workflow element is likely to return several entries with the same DN.

To eliminate duplicate entries from search operations:

Add the `GetRidOfDuplcate` workflow element before any workflow element, such as the Join workflow element, that returns duplicate entries.

The following example creates a `get-rid-of-duplicate WFE (next WFE=NEXT_WFE).`

```
dsconfig create-workflow-element \
          --set enabled:true \
          --set next-workflow-element:NEXT_WFE \
           --set cache-size:1000000 \
          --type get-rid-of-duplicate \
          --element-name example \
          --hostname locahost \
          --port 1444 \
          -X \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile ****** \
          --no-prompt
```

In this example, a search will return no more than 1000000 unique entries.

> **Note:**
>
> In this configuration example, the created workflow element is not part of any workflow chain. A full configuration must also define or create the workflow chain, and update the Network group.

The `GetRidofDuplicate` has one configuration parameter:

`cache-size`

The `cache-size` parameter is required. It specifies the maximum number of entries that can be returned to the client during a single search operation.

24.2.2 Filtering Search Results Using the `HideByFilter` Workflow Element

The `HideByFilter` workflow element enables you to control in fine detail which entries are returned by searches of a virtual directory.

For example, if you are using Oracle Unified Directory as an address book directory, you can display only the entries for customer service representatives. First you give all customer service representatives an `ou` value of `CSR`. Then can use the `HideByFilter` workflow element with `hideFilter` set to `ou=CSR`. When the directory is searched, only the customer service representatives entries are returned.

To filter search results using the HideByFilter workflow element:

Create and link a `HideByFilter` workflow element. For example:

```
dsconfig create-workflow-element \
          --set enabled:true \
          --set next-workflow-element:NEXT_WFE \
          --set ldap-filter:ou=CSR \
          --type hide-entries-by-filter \
          --element-name example1 \
          --hostname dosapano \
          --port 1444 \
          -X \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile ****** \
          --no-prompt
```

Table 24-1 summarizes the HideByFilter plug-in configuration parameters:

Table 24-1 HideByFilter Parameters

Parameter	Description
`hideFilter`	• Static Filter Example: If hideFilter = (department=Sales) then only entries with the attribute department=Sales are returned to the client application. • Dynamic Filter Example: If hideFilter = (department=%department%) then %department% is replaced with the department attribute value of the bound user.
`ldapURL` (multivalued)	If an entry matches the ldapURL filter then it is returned to the client application only if it's a descendant of the LdapURL base DN. All the other fields of the LDAP filter are ignored.

Table 24-1 (Cont.) HideByFilter Parameters

Parameter	Description
`adapterNames`	A list of adapters from which the user entry for the dynamic filter is searched. If list is empty, or if the user entry can be found in none of the adapters (including the current adapter) then the dynamic filter is ignored.
`applyForAdmin`	When set to true, the filtering does apply to admin users. The parameter is optional and the default value is false.

24.3 Adding the `memberof` User Attribute to `person` Entries

You can add the `memberof` user attribute to `person` entries. This is useful when you want applications to see group membership, but do not want them to perform secondary searches for those groups.

> **Note:**
>
> For more information, see Understanding Addition of `memberof` User Attributes to `person` Entries.

To define a VirtualMemberof workflow element, use the following configuration parameters:

- **searchBase**: DN of the base to search for groups containing `person` entries.
- **explicitRequestOnly**: Specify `True` or `False`
  - **True** (*default*): Adds the `memberof` attribute to the entry only if it is explicitly requested as a returned attribute.
  - **False**: Always adds the `memberof` attribute to the entry.
- **member-attribute-name**: The name of the `memberof` attribute to add.

> **Note:**
>
> The `memberof` attribute has a default value for Oracle Virtual Directory convergence.
>
> In Oracle Virtual Directory, the `memberof` attribute is a `user` attribute (not operational). The definition is:
>
> ```
> attributeTypes: ( 1.2.840.113556.1.2.102 NAME 'memberOf'
>   DESC 'The distinguished name of the groups to which this object belongs'
>   EQUALITY distinguishedNameMatch
>   SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
>   X-ORIGIN 'Microsoft Active Directory' )
> ```

24.4 Performing DN Renaming

You can perform DN Renaming configuration. It can be viewed and modified using dsconfig commands.

The following topics describe the DN Renaming configuration:

- Configuring DN Renaming
- Creating a DN Renaming Workflow Element
- Modifying a DN Renaming Configuration

24.4.1 Configuring DN Renaming

To configure DN renaming, you must first create a DN renaming workflow element and then you can modify the DN renaming properties.

You can modify the following DN renaming properties:

- `client base DN`
- `source base DN`
- `next workflow element`
- `black list attributes`
- `white list attributes`

24.4.2 Creating a DN Renaming Workflow Element

To create a DN renaming workflow element, use the `dsconfig create-workflow-element` command.

Follow the below given instructions:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
 create-workflow-element \
 --type dn-renaming \
 --element-name RenameorgDN \
 --set client-base-dn:ou=myorg,dc=example,dc=com \
 --set next-workflow-element:load-bal-we1 \
 --set source-base-dn:ou=people,dc=example,dc=com \
 --set enabled:true
```

where:

- `--set client-base-dn` indicates the client base DN, which is the workflow entry point
- `--set source-base-dn` indicates the base DN which the entries should have after transformation, which is the workflow exit point.
- `--set next-workflow-element` indicates the workflow element that will follow the DN renaming workflow element in the proxy architecture. You can specify any type of workflow element here.

24.4.3 Modifying a DN Renaming Configuration

You can view and modify a DN renaming configuration by using the `dsconfig` commands.

- To view the current DN renaming properties, use the `dsconfig get-workflow-element-prop` command.
- To modify a DN renaming property, use the `dsconfig set-workflow-element-prop` command. For example,

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
 set-workflow-element-prop \
 --element-name RenameorgDN \
 --set source-base-dn:ou=admin,dc=example,dc=com
```

  In the preceding example, only the `source-base-dn` is modified. There is no need to specify the old source base DN. Only the new one is required.

- To create a black list of DN attributes that should not be renamed by using, use the `dsconfig set-workflow-element-prop` command. For example,

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-workflow-element-prop --element-name RenameorgDN \
  --set black-list-attributes:manager
```

  The attribute must have a DN type.

24.5 Performing RDN Changing Configuration

You can perform RDN changing configuration, create RDN changing workflow element using `dsconfig create-workflow-element` command and modify RDN values.

The following topics describe how to perform RDN changing configuration:

- Configuring RDN Changing
- Creating an RDN Changing Workflow Element
- Modifying RDN Values

24.5.1 Configuring RDN Changing

To change RDNs, you must first create an RDN Changing workflow element, and then you can modify the properties.

Modify the below given required properties:

- `client RDN`
- `source RDN`
- `next workflow element`
- `objectclass`
- `dn attributes`
- `replace-value`

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

24.5.2 Creating an RDN Changing Workflow Element

To create an RDN Changing workflow element, use the `dsconfig create-workflow-element` command.

Use the following commands to create an RDN changing workflow element:

```
dsconfig create-workflow-element \
          --set client-rdn:cn \
          --set enabled:true \
          --set next-workflow-element:localproxy \
          --set source-rdn:uid \
          --type rdn-changing \
          --element-name myrdnchangingwfe \
          --hostname localhost \
          --port "4444" \
          --trustAll \
          --bindDN cn=directory\ manager \
          --bindPasswordFile pwd-file \
          --no-prompt
```

where:

- `--set client-rdn` indicates the client base RDN, which is the workflow entry point.
- `--set source-rdn` indicates the base RDN which the entries should have after transformation, which is the workflow exit point.
- `--set next-workflow-element:localproxy` indicates the workflow element that will follow the RDN changing workflow element in the proxy architecture. This can be any type of workflow element.

  > **Note:**
  >
  > You must create the Proxy LDAP workflow element with the parameters
  >
  > – `remote-root-dn`
  >
  > – `remote-root-password`
  >
  > The RDN Changing workflow element uses these credentials to perform internal searches on the remote server.

- `--element-name myrdnchangingwfe` indicates the name of the RDN Changing workflow element you are creating.

  This configuration replaces `uid=user.1,ou=people,dc=example,dc=com with cn=User CN,ou=people,dc=example,dc=com`.

24.5.3 Modifying RDN Values

After you have configured an RDN changing workflow element, you can view and modify RDN values by using `dsconfig` commands.

1. To view the current RDN properties, use the `dsconfig get-workflow-element-prop` command.
2. To rename or replace an RDN property, use the `dsconfig set-workflow-element-prop` command.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
 set-workflow-element-prop \
 --element-name myrdnchangingwfe \
 --set source-rdn:uid
```

In the preceding example, only the `source-rdn` is modified. There is no need to specify the old `source-rdn`. Only the new one is required.

24.6 Configuring Transformations

You can configure transformations by using `dsconfig` and OUDSM.

The following topics explain the transformations configuration model:

- Understanding the Configuration Model
- Configuring Transformation Using `dsconfig`
- Configuring Transformations Using OUDSM

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

> **Note:**
>
> For more information about transformations, see Understanding the Transformation Framework.

24.6.1 Understanding the Configuration Model

The transformation workflow element and transformations are the backbone entities for configuring transformation.

The *transformation workflow element* is a container that contains a list of references to *transformations*. One transformation can be reused by multiple transformation workflow elements. *Conditions* are properties (attributes) that you can set either on a transformation workflow element or on a transformation.

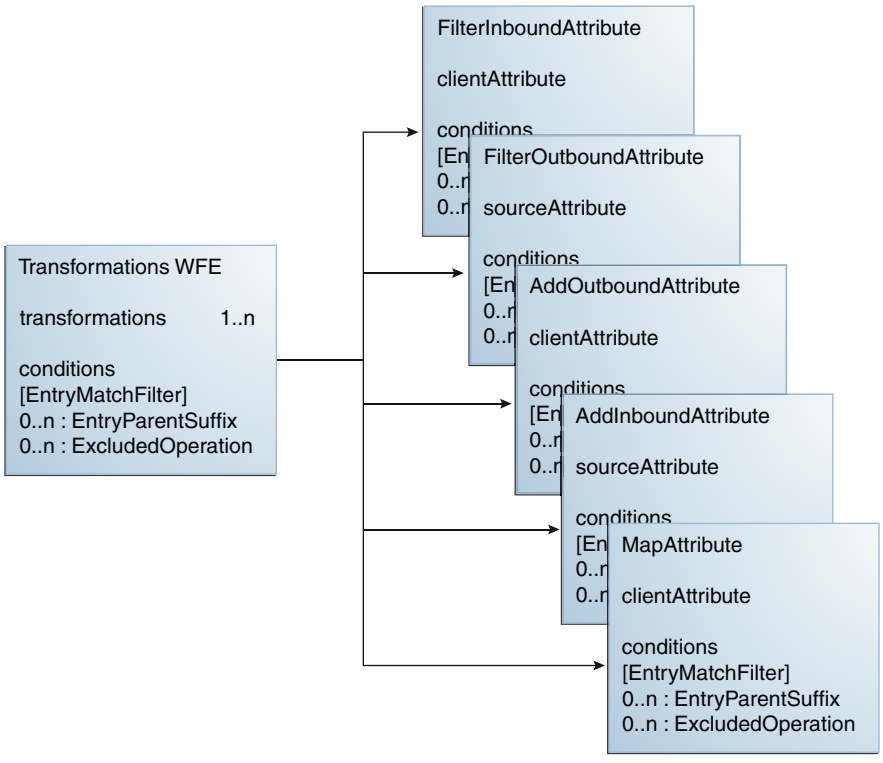
> **Note:**
>
> For detailed information about the various transformation types, conditions, and parameters that you can configure for a transformation workflow element, see Components of Transformation.

You cannot configure the order in which the transformations should work. For example, you define a transformation workflow element that uses transformation A and transformation B. But, you cannot determine if an entry is first processed by transformation A and then by transformation B. It can be B before A.

If you must define the order in which transformations should occur, for example transformation A should happen before transformation B, then it is recommended that you first create a transformation workflow element that uses transformation A. Next, create another transformation workflow element that uses transformation B. Then, place the second transformation workflow element after the first transformation workflow element.

Figure 24-1 illustrates a high-level configuration model.

Figure 24-1 Configuration Model



24.6.2 Configuring Transformation Using `dsconfig`

You can create transformations, a workflow element, add transformations, and associate conditions using the `dsconfig` CLI.

> **Note:**
>
> - To create a transformations workflow element using OUDSM, see Configuring Transformations Using OUDSM.
> - For more detailed information about transformations, transformation workflow elements, and conditions, see Understanding the Transformation Framework,

To configure transformation:

1. Create a first transformation of type `filter-outbound-attribute`.

```
$ dsconfig create-transformation -X -n -Q -p -D cn="directory manager" -j pwd-file \
--set source-attribute:description \
--type filter-outbound-attribute\
--transformation-name fodescription
```

2. Create another transformation of type `add-outbound-attribute`.

```
$ dsconfig create-transformation -X -n -Q -p -D cn="directory manager" -j pwd-file
\
--set client-attribute:legacyemail=%cn%.%sn%@mycompany.com \
--type add-outbound-attribute \
--transformation-name legacyemail
```

3. Create the `transformations-workflow-element` with the first transformation, and add it to the processing flow.

```
$ dsconfig create-workflow-element -X -n -Q -p -D cn="directory manager" -j pwd-file
\
--set transformation:legacyemail \
--set set next-workflow-element:pxywfe \
--type transformations \
--element-name trsfwfe

$ sdsconfig set-workflow-prop -X -n -Q -p -D cn="directory manager" -j pwd-file \
--workflow-name pxywf \
--set workflow-element:trsfwfe
```

4. Add the second transformation to the workflow element.

```
$ dsconfig set-workflow-element-prop -X -n -Q -p -D cn="directory manager" -j pwd-
file \
--element-name trsfwfe \
--add transformation:fodescription
```

5. Define the transformation criteria, which is that the transformation will occur only under `cn=users`.

```
$ dsconfig set-workflow-element-prop -X -n -Q -p -D cn="directory manager" -j pwd-
file \
--element-name trsfwfe \
--set entry-parent-suffix:cn=users,dc=example
```

6. Set that transformations will happen only for users located in Paris.

```
$ dsconfig set-workflow-element-prop -X -n -Q -p -D cn="directory manager" -j pwd-
file \
--element-name trsfwfe \
--set entry-match-filter:l=Paris
```

7. Create a new mapping transformation and add it to the workflow element.

```
$ dsconfig create-transformation -X -n -Q -p -D cn="directory manager" -j pwd-file
\
--set client-attribute:faxnum=%facsimileTelephoneNumber% \
--type map-attribute \
--transformation-name mapfax

$ dsconfig set-workflow-element-prop -X -n -Q -p -D cn="directory manager" -j pwd-
file \
--element-name trsfwfe \
--add transformation:mapfax
```

8. Set that this transformation will happen only for persons.

```
$ dsconfig set-transformation-prop -X -n -Q -p -D cn="directory manager" -j pwd-file
\
--transformation-name mapfax \
--set entry-match-filter:\(objectclass=person\)
```

24.6.3 Configuring Transformations Using OUDSM

You can create, modify, and delete a transformation workflow element for Oracle Unified Directory proxy servers using OUDSM.

> **Note:**
>
> To create a transformation workflow element using `dsconfig`, see Configuring Transformation Using `dsconfig`.

This section includes the following topics:

- Creating Transformations
- Modifying Transformations
- Deleting Transformations
- Selecting Values from Value Definition Screen

24.6.3.1 Creating Transformations

If you are connected to an Oracle Unified Directory Proxy Server, then OUDSM allows you to create five different types of transformations.

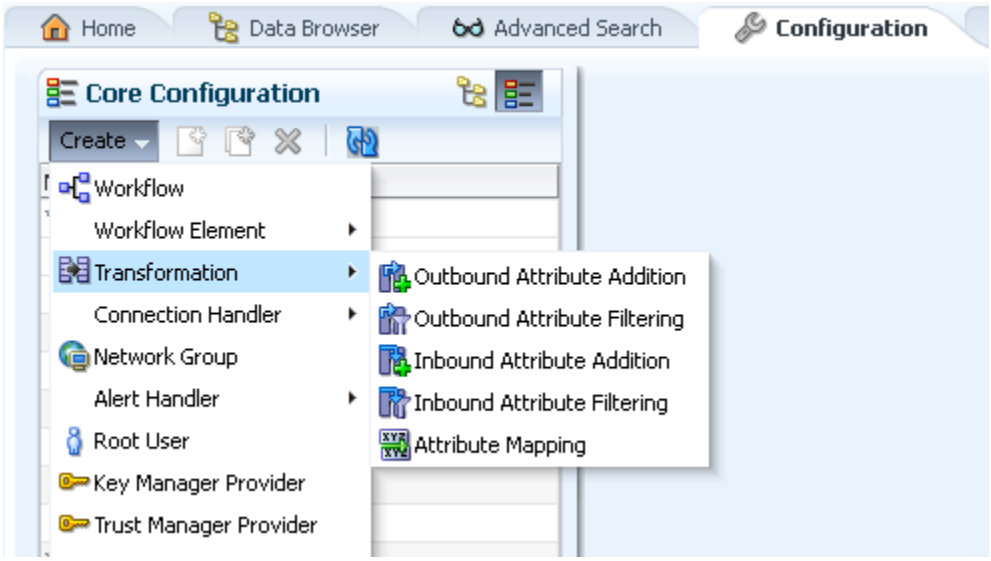
For more information about the types of transformations supported, see Overview of Transformation Types.

> **Note:**
>
> If you are connected to an Oracle Unified Directory server instance, then the option to create a new Transformation is not available because transformation functionality is supported by proxy servers only.

To create a transformation using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.
4. From the **Create** menu, select **Transformation.**
5. From the **Transformation** submenu, select the desired transformation type.

Figure 24-2 Transformation Types



In this example, consider the following properties for an Outbound Attribute Addition transformation type.

> **Note:**
>
> The properties that appear while creating a transformation vary depending on the type of transformation you create. For more information about each transformation type and the associated properties, see Overview of Transformation Types.

6. In the **Name** field, type the name for the transformation.
7. In the Conditions region, enter the following information:

> **Note:**
>
> Conditions are optional. However; at runtime, conditions specified here at the transformation level are used with those specified at the transformation workflow element level in the transformation workflow element where the transformation is used. For more information about transformation workflow element, see Configuring Workflow Elements Using OUDSM.

a. In the **Entry Matching Filter** field, type a valid LDAP filter.

b. In the **Entry Parent Suffixes** box, click **Add** to specify the DN that must be an ascendant.

   To select an entry, click **Select.**

   In the **Entry Picker** window, select **Tree View** to navigate the directory tree and locate the entry, or **Search View** to search for the entry.

c. From the **Excluded Operations** list, select the operations that you want to exclude.

8. In the **Transformation Definition** region, enter the following information:

   a. In the **Client Attribute** field, type the name of the client virtual attribute.

      To select a client attribute entry, click **Select.**

      In the **Attribute Picker** window, select locate the desired entry, or Click **Search** to search for the entry.

   b. In the **Value Definitions** box, click **Add** to specify the value definitions of the client virtual attribute.

      Click **Define** to enter an appropriate value definition. For more information about specifying value definitions, see Selecting Values from Value Definition Screen.

9. From the **Conflict Behavior** list, select the desired conflict behavior policy.
10. Click **Virtual in Source** to **Yes.**
11. Click **Create.**

24.6.3.2 Modifying Transformations

This section describes how to modify the properties for a transformation.

In this example, modify the properties for an Outbound Attribute Addition transformation type created in Creating Transformations.

To modify a transformation:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.
4. Expand the **Transformations** element.
5. Click the desired transformation.

   Transformation configuration details appear for modification in the right pane.
6. Modify the required information.
7. Click **Apply.**

24.6.3.3 Deleting Transformations

This section describes the procedure to delete Transformation using OUDSM.

To delete a transformation:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.

2. Select the **Configuration** tab.
3. Select the **Core Configuration** view.
4. Expand the **Transformations** element.
5. Select the desired transformation to delete.

   The Delete configuration window appears seeking confirmation before deleting.
6. Click **OK.**

24.6.3.4 Selecting Values from Value Definition Screen

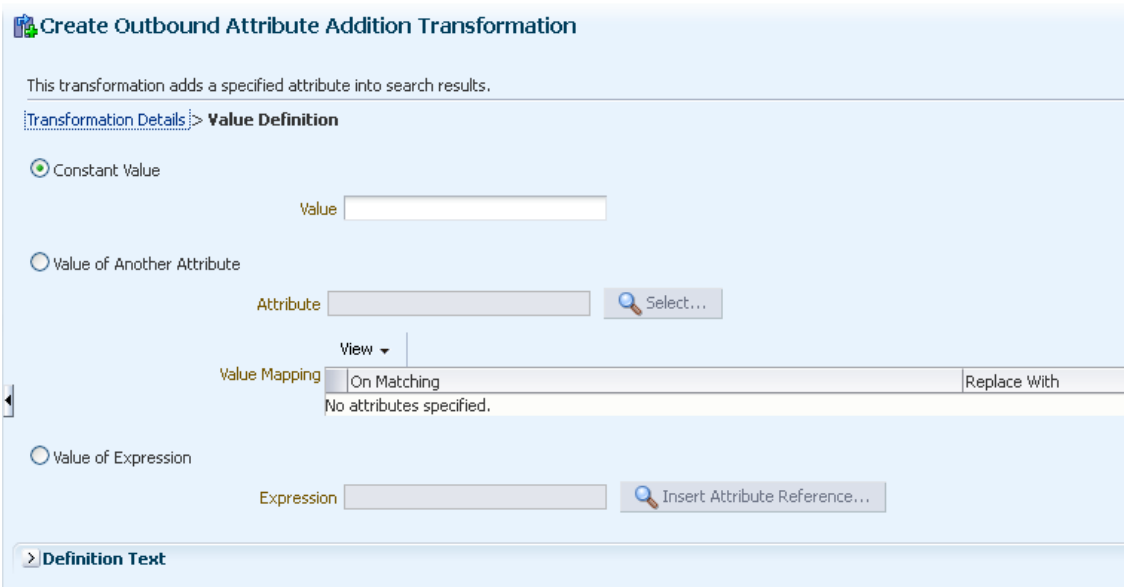
The Value Definition Builder subscreen allows you to define a value for an attribute that is being added, mapped, or deleted by a transformation.

You can specify the following values:

- Constant value: It is used to enter a constant value.
- Value of another attribute: It is used to create a new attribute from an existing attribute in the entry that is being processed or to filter a value taken from another attribute.
- Value of expression: It is used to create an attribute value or to filter an attribute value by manipulating the value of one or more existing attributes.

Figure 24-3 shows the Value Definition screen.

Figure 24-3 Value Definition Screen



24.7 Configuring SAML XASP

The `dsconfig` command allows you to create a new SAML XASP workflow element and also edit the properties of an existing workflow element.

- Creating a New SAML XASP Workflow Element Using the dsconfig Command
- Modifying the Properties of an Existing SAML XASP Workflow Element

24.7.1 Creating a New SAML XASP Workflow Element Using the dsconfig Command

The `dsconfig create-workflow-element --type saml-xasp` command allows you to create new SAML XASP workflow elements.

To create a new SAML XASP workflow element:

- Run the `dsconfig create-workflow-element --type saml-xasp` command:

```
$ ./dsconfig create-workflow-element \
    --type saml-xasp \
    --set enabled:true \
    --set java-
class:org.opends.server.workflowelement.ovdplugin.xasp.SamlXaspWorkflowElement \
    --element-name test01 \
    --set xasp-attribute-name:certificatedn \
    --set xasp-base-dn:dc=example,dc=com \
    --set xasp-contains-dn:dc=example,dc=com \
    --set xasp-ws-url:http://host01.example.com:7777/fed/ar/soap
    --hostname host01.example.com \
    --port 6444
    --bindDN "cn=Directory Manager" \
    --bindPasswordFile /home/oracle/pwd.txt \


>>>> Configure the properties of the Saml Xasp Workflow Element

          Property             Value(s)
          ---------------------------------------------------------------------
    1)    enabled              true
    2)    java-class           org.opends.server.workflowelement.ovdplugin.xasp.
                               SamlXaspWorkflowElement
    3)    xasp-attribute-name  certificatedn
    4)    xasp-base-dn         "dc=example,dc=com"
    5)    xasp-contains-dn     "dc=example,dc=com"
    6)    xasp-debug           false
    7)    xasp-index           -
    8)    xasp-response        -
    9)    xasp-reverse-dn      false
    10)   xasp-ttl             -
    11)   xasp-ws-url          http://host01.example.com:7777/fed/ar/soap

    ?)    help
    f)    finish - create the new Saml Xasp Workflow Element
    q)    quit

Enter choice [f]:

The Saml Xasp Workflow Element was created successfully
```

For more information on the configuration properties of the SAML XASP workflow element, see Configuration Parameters for SAML XASP Workflow Element.

24.7.2 Modifying the Properties of an Existing SAML XASP Workflow Element

The `dsconfig set-workflow-element-prop` command allows you to edit the properties of an existing SAML XASP workflow element.

To modify an existing SAML XASP property:

- Run the `dsconfig set-workflow-element-prop` command:

```
$ ./dsconfig set-workflow-element-prop \
    --element-name test01 \
    --set xasp-attribute-name:certificatedn100 \
    --hostname host01.example.com \
    --port 6444
    --bindDN "cn=Directory Manager" \
    --bindPasswordFile /home/oracle/pwd.txt \


>>>> Configure the properties of the Saml Xasp Workflow Element

          Property             Value(s)
          ---------------------------------------------------------------------
    1)    enabled              true
    2)    java-class           org.opends.server.workflowelement.ovdplugin.xasp.
                               SamlXaspWorkflowElement
    3)    xasp-attribute-name  certificatedn100
    4)    xasp-base-dn         "dc=example,dc=com"
    5)    xasp-contains-dn     "dc=example,dc=com"
    6)    xasp-debug           false
    7)    xasp-index           -
    8)    xasp-response        -
    9)    xasp-reverse-dn      false
    10)   xasp-ttl             -
    11)   xasp-ws-url          http://host01.example.com:7777/fed/ar/soap

    ?)    help
    f)    finish - apply any changes to the Saml Xasp Workflow Element
    q)    quit

Enter choice [f]:


The Saml Xasp Workflow Element was modified successfully
```

> **Note:**
>
> In the preceding example, only the `xasp-attribute-name` property is modified. There is no need to specify the old XASP attribute name. Only the new one is required.

24.8 Deploying ForkJoin Workflow Element Configuration Model

The `dsconfig` command allows you to create and configure a ForkJoin workflow element.

Topics

24.8.1 Understanding ForkJoin Workflow Element Configuration Model

Consider a scenario, where you have two directory servers namely `oud1` and `oud2`. Here, `oud1` is the primary participant and `oud2` is the secondary participant. Data resides in the both the primary participant and the secondary participant.

For this scenario, assume the following:

- The primary participant namespace is `dc=example,dc=com`.
- The secondary participant namespace is `dc=example,dc=com`.
- The ForkJoin workflow element suffix is `dc=forkjoin`.

Before creating the ForkJoin workflow element, you must configure the participating workflow elements so that you can link to them from the ForkJoin workflow element configuration. For each directory, you must create a Proxy LDAP workflow element that is associated with a directory to retrieve information from that directory. For example, consider a scenario with two separate Proxy LDAP workflow elements:

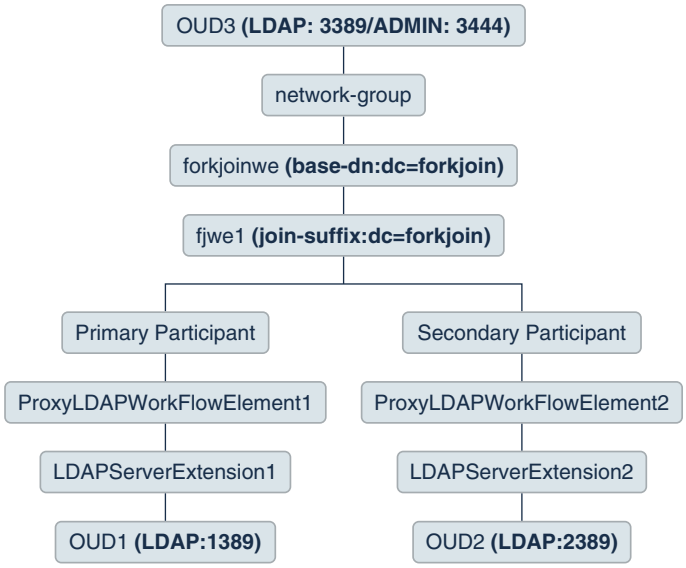
- The first Proxy LDAP workflow element, `ProxyLDAPWorkFlowElement1` , is linked to the primary participant of the ForkJoin workflow element configuration.
- The second Proxy LDAP workflow element, `ProxyLDAPWorkFlowElement2`, is linked to the secondary participant of the ForkJoin workflow element configuration.

> **Note:**
>
> You can also configure an RDBMS workflow element as a primary or a secondary participant.

Assume `oud3` is a proxy workflow element, which has a ForkJoin workflow element pointing to the preceding participants through `ProxyLDAPWorkFlowElement1` (to `oud1`) and `ProxyLDAPWorkFlowElement2` (to `oud2`). To learn how to deploy the ForkJoin workflow element configuration, see Implementing ForkJoin Workflow Element Configuration Model.

The following diagram provides a pictorial representation of the ForkJoin workflow element configuration model.

Figure 24-4 ForkJoin Workflow Element Configuration Model



The `secondary-only-attributes` parameter is set to `title` in ForkJoin workflow element and the `join-rule` is set as `cn=cn`. Data in secondary participant as mentioned in the following table, does not have the `description` attribute.

The following table lists the data that resides in the primary participant and secondary participant.

Table 24-2 Data in Primary Participant and Secondary Participant

Data in Primary Participant	Data in Secondary Participant
dn: cn=Rock,dc=example,dc=com objectclass: inetorgperson cn: Rock sn: Anne givenname: Anne rock telephonenumber: 54300	dn: cn=Rock,dc=example,dc=com objectclass: inetorgperson cn: Rock sn: Anne title: Manager
dn: cn=Sandy,dc=example,dc=com objectclass: inetorgperson cn: Sandy sn: Ketty manager: cn=Rock, dc=primary telephonenumber: 54301	dn: cn=Sandy,dc=example,dc=com objectclass: inetorgperson cn: Sandy sn: Ketty title: SMTS

Table 24-2 (Cont.) Data in Primary Participant and Secondary Participant

Data in Primary Participant	Data in Secondary Participant

`dn: cn=Rivry,dc=example,dc=com`
`objectclass: inetorgperson`
`cn: Rivry`
`sn: Rod`
`title: Trainee`
`manager: cn=Rock, dc=secondary`
`telephonenumber: 54303`
`description: Trainee for dept 543`
`departmentNumber: 543`
	`dn: cn=Rivry,dc=example,dc=com`
	`objectclass: inetorgperson`
	`cn: Rivry`
	`sn: Rod`
	`title: Trainee`

`dn: cn=Woods,dc=example,dc=com`
`objectclass: inetorgperson`
`cn: Woods`
`sn: Tent`
`description: User with no title`
	`dn: cn=Mounty,dc=example,dc=com`
	`objectclass: inetorgperson`
	`cn: Mounty`
	`sn: Ret`
	`title: MTS - dept_sec`

24.8.2 Implementing ForkJoin Workflow Element Configuration Model

You can create and configure a ForkJoin workflow element to aggregate data from two data sources at real time by using `dsconfig` command.

- Preparing For ForkJoin Workflow Element Configuration
- Configuring OUD Proxy Server For ForkJoin Workflow Element Configuration
- Creating ForkJoin Workflow Element
- Configuring ForkJoin Workflow Element
- Configuring ForkJoin Workflow Element Join Policy
- Validating ForkJoin Workflow Element Configuration

> **Note:**
>
> See Understanding ForkJoin Workflow Element Configuration Model to comprehend the ForkJoin workflow element configuration model.

24.8.2.1 Preparing For ForkJoin Workflow Element Configuration

To deploy a ForkJoin workflow element configuration, you need to set up the OUD Directory Server instance and the OUD Proxy Server instance. You need to place a proxy server in front of the data sources that you want to join.

Set up the First OUD Instance (`oud1`)

1. Run the `oud-setup` command to create the `oud1` instance as follows:

```
./oud-setup --cli --baseDN "dc=example,dc=com" --addBaseEntry --
adminConnectorPort 1444 --ldapPort 1389 \
--rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt --no-
prompt --noPropertiesFile
```

2. Populate the `oud1` directory server instance with sample entries.
   a. Create an LDIF file (`fj_oud1.ldif`) with Data in Primary Participant as described in Understanding ForkJoin Workflow Element Configuration Model.
   b. Run the `ldapmodify` command to populate the `oud1` instance with the entries in `fj_oud1.ldif` file.

```
./ldapmodify --hostname host01.example.com --port 1389 --bindDN
"cn=Directory Manager" --bindPasswordFile pwd.txt --defaultAdd --
filename fj_oud1.ldif
```

Set up the Second OUD Instance (oud2)

1. Run the `oud-setup` command to create the `oud2` instance as follows:

```
./oud-setup --cli --baseDN "dc=example,dc=com" --addBaseEntry --
adminConnectorPort 2444 --ldapPort 2389 \
--rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt --no-
prompt --noPropertiesFile
```

2. Populate the `oud2` directory server instance with sample entries.
   a. Create an LDIF file (`fj_oud2.ldif`) with Data in Secondary Participant as described in Understanding ForkJoin Workflow Element Configuration Model.
   b. Run the `ldapmodify` command to populate the `oud2` instance with the entries in `fj_oud2.ldif` file.

```
./ldapmodify --hostname host01.example.com --port 2389 --bindDN
"cn=Directory Manager" --bindPasswordFile pwd.txt --defaultAdd --
filename fj_oud2.ldif
```

Set up the OUD Proxy Server Instance (oud3)

1. Run the `oud-proxy-setup` command to create a proxy server instance, `oud3`, as follows:

```
./oud-proxy-setup --cli --ldapPort 3389 --adminConnectorPort 3444 --
rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt
```

24.8.2.2 Configuring OUD Proxy Server For ForkJoin Workflow Element Configuration

To connect to a remote LDAP directory server, the Oracle Unified Directory proxy needs LDAP server extension and LDAP proxy workflow element.

LDAP Server extensions are the properties required to connect from OUD Proxy `oud3` to the remote LDAP servers (`oud1` and `oud2`). You create LDAP Server extensions for `oud1` and `oud2`. You will use these extensions in the workflow configuration later.

In addition, you need to create proxy workflow elements for `oud1` and `oud2`. These specify connection details and credentials to the remote LDAP servers.

1. Create an LDAP Server Extension (`LDAPServerExtension1`) and that points to `oud1`.

```
./dsconfig create-extension \
        --set enabled:true \
        --set remote-ldap-server-address:host01.example.com \
        --set remote-ldap-server-port:1389 \
        --type ldap-server \
        --extension-name LDAPServerExtension1 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

2. Create an LDAP Proxy workflow element (`ProxyLDAPWorkFlowElement1`) that points to `oud1`.

```
./dsconfig create-workflow-element \
        --set client-cred-mode:use-client-identity \
        --set enabled:true \
        --set ldap-server-extension:LDAPServerExtension1 \
        --type proxy-ldap \
        --element-name ProxyLDAPWorkFlowElement1 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

3. Create an LDAP Server Extension (`LDAPServerExtension2`) that points to `oud2`.

```
./dsconfig create-extension \
        --set enabled:true \
        --set remote-ldap-server-address:host01.example.com \
        --set remote-ldap-server-port:2389 \
        --type ldap-server \
        --extension-name LDAPServerExtension2 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

4. Create an LDAP Proxy workflow element (`ProxyLDAPWorkFlowElement2`) that points to `oud2`.

```
./dsconfig create-workflow-element \
        --set client-cred-mode:use-client-identity \
        --set enabled:true \
```

```
--set ldap-server-extension:LDAPServerExtension2 \
--type proxy-ldap \
--element-name ProxyLDAPWorkFlowElement2 \
--hostname host01.example.com \
--port 3444 \
--trustAll \
--bindDN cn=Directory\ Manager \
--bindPasswordFile pwd.txt \
--no-prompt
```

5. Run the `dsconfig` command to view the server extensions.

```
./dsconfig -h host01.example.com -p 3444 -D "cn=Directory Manager" --
bindPasswordFile pwd.txt -X -n list-extensions
```

6. Run the `dsconfig` command to view the proxy LDAP workflow elements.

```
./dsconfig -h host01.example.com -p 3444 -D "cn=Directory Manager" --
bindPasswordFile pwd.txt -X -n list-workflow-elements
```

24.8.2.3 Creating ForkJoin Workflow Element

You can create a ForkJoin workflow element using `dsconfig` command.

1. Create workflow element of type `fork-join` for join-suffix `dc=forkjoin`.

```
./dsconfig create-workflow-element \
          --set enabled:true \
          --set join-suffix:dc=forkjoin \
          --set populate-joinedentrydn:true \
          --set secondary-only-attributes:title \
          --type fork-join \
          --element-name fjwe1 \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

2. Create a workflow of type `generic` for join-suffix `dc=forkjoin`.

```
./dsconfig create-workflow \
          --set base-dn:dc=forkjoin \
          --set enabled:true \
          --set workflow-element:fjwe1 \
          --type generic \
          --workflow-name forkjoinwf \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
```

```
            --bindPasswordFile pwd.txt \
            --no-prompt
```

24.8.2.4 Configuring ForkJoin Workflow Element

You can configure the ForkJoin workflow element using `dsconfig` command.

1. Create the Primary ForkJoin participant that is the link between the ForkJoin workflow element and `oud1`.

```
./dsconfig create-primary-fork-join-participant \
          --element-name fjwe1 \
          --set participant-dn:dc=example,dc=com \
          --set participating-workflow-element:ProxyLDAPWorkFlowElement1 \
          --type generic \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

2. Create the Secondary ForkJoin participant that is the link between the ForkJoin workflow element and `oud2`.

```
./dsconfig create-secondary-fork-join-participant \
          --element-name fjwe1 \
          --set participant-dn:dc=example,dc=com \
          --set participating-workflow-element:ProxyLDAPWorkFlowElement2 \
          --set participants-join-rule:cn=cn \
          --type generic \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

3. Configure the `secondary-only-attributes` property for the ForkJoin workflow element.

```
./dsconfig --hostname host01.example.com  --port 3444  --trustAll  --
bindDN "cn=Directory Manager" \
--bindPasswordFile pwd.txt --no-prompt set-workflow-element-prop --element-
name fjwe1 --add secondary-only-attributes:description
```

4. Attach the ForkJoin Workflow element (`forkjoinwf`) to the network group.

```
./dsconfig set-network-group-prop \
          --group-name network-group \
          --set workflow:forkjoinwf \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
```

```
            --trustAll \
            --bindDN cn=Directory\ Manager \
            --bindPasswordFile pwd.txt \
            --no-prompt
```

24.8.2.5 Configuring ForkJoin Workflow Element Join Policy

ForkJoin workflow element supports `standard-join`, `left-outer-join`, and `full-outer-join` Join policies. Learn to configure the Join policy.

1. Set the join-policy parameter to `full-outer-join`.

```
./dsconfig set-secondary-fork-join-participant-prop \
          --element-name fjwe1 \
          --set join-policy:full-outer-join \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

2. Set the join-policy parameter to `standard-join`.

```
./dsconfig set-secondary-fork-join-participant-prop \
          --element-name fjwe1 \
          --set join-policy:standard-join \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

3. Set the join-policy parameter to `left—outer-join`.

```
./dsconfig set-secondary-fork-join-participant-prop \
          --element-name fjwe1 \
          --set join-policy:left-outer-join \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

24.8.2.6 Validating ForkJoin Workflow Element Configuration

Learn to validate the ForkJoin workflow element configuration.

1. To test the `full-outer-join` condition, run the `ldapsearch` command as follows:

```
./ldapsearch -h host01.example.com -p 3389 -D "cn=Directory Manager" -j
pwd.txt -b "dc=forkjoin" -s sub "|(sn=*e*)(title=*e*)" sn cn title


cn=Rock,dc=forkjoin
sn=Anne
cn=Rock
title=Manager

cn=Sandy,dc=forkjoin
sn=Ketty
cn=Sandy
title=SMTS

cn=Woods,dc=forkjoin
sn=Tent
cn=Woods

cn=Rivry,dc=forkjoin
sn=Rod
cn=Rivry
title=Trainee

cn=Mounty,dc=forkjoin
sn=Ret
cn=Mounty
title=MTS - dept_sec
```

2. To test the `standard-join` condition, run the `ldapsearch` as follows:

```
./ldapsearch -h host01.example.com -p 3389 -D "cn=Directory Manager" -j
pwd.txt -b "dc=forkjoin" -s sub "|(sn=*e*)(title=*e*)" sn cn title


cn=Rock,dc=forkjoin
sn=Anne
cn=Rock
title=Manager

cn=Sandy,dc=forkjoin
sn=Ketty
cn=Sandy
title=SMTS

cn=Woods,dc=forkjoin
sn=Tent
cn=Woods
```

3. To test the `left-outer-join` condition, run the `ldapsearch` as follows:

```
./ldapsearch -h host01.example.com -p 3389 -D "cn=Directory Manager" -j
pwd.txt -b "dc=forkjoin" -s sub "|(sn=*e*)(title=*e*)" sn cn title


cn=Rock,dc=forkjoin
sn=Anne
cn=Rock
title=Manager

cn=Sandy,dc=forkjoin
sn=Ketty
cn=Sandy
title=SMTS

cn=Woods,dc=forkjoin
sn=Tent
cn=Woods

cn=Rivry,dc=forkjoin
sn=Rod
cn=Rivry
title=Trainee
```

24.9 Configuring DynamicGroup Workflow Element

You can configure DynamicGroup Workflow Element by using `dsconfig`.

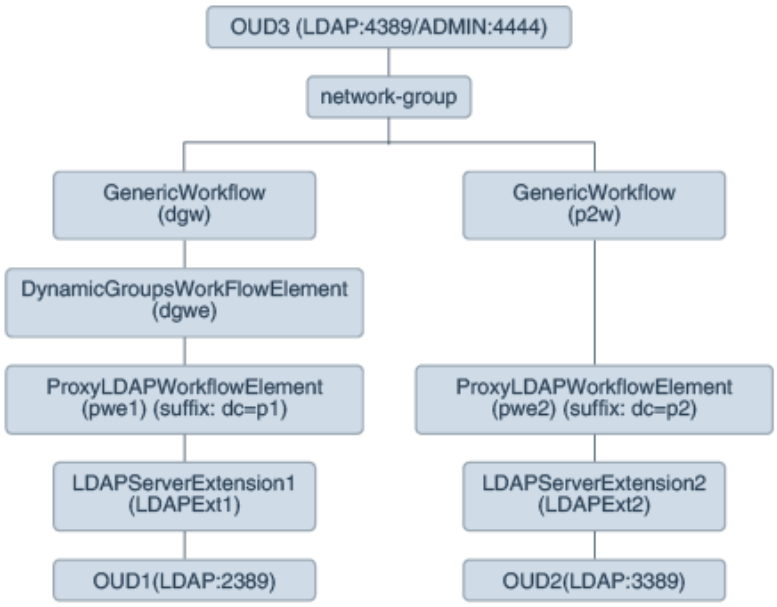
The following sections describe configuring dynamic group workflow element:

- Understanding DynamicGroup Workflow Element Configuration Model
- Implementing DynamicGroup Workflow Element Configuration Model
- Testing the DynamicGroup Workflow Element Configuration

24.9.1 Understanding DynamicGroup Workflow Element Configuration Model

Learn about the DynamicGroup Workflow Element Configuration.

The following diagram illustrates the DynamicGroup Workflow Element configuration:



In the above example, for each backend directory, a `ProxyLDAPWorkflowElement` is created that retrieves information from the backend directory. Depending on the bind DN, a search would be routed through `network-group` and forwarded to `workflow` elements `dgw` or `p2w`.The `DynamicGroupsWorkflowElement` does the conversation with the directory servers that are dependent on the Proxy LDAP Workflow elements.

In this scenario, assume the following:

- `oud1` is associated with a `ProxyLDAPWorkflowElement` `pwe1` with suffix `dc=p1`.
- `oud2` is associated with `ProxyLDAPWorkflowElement` `pwe2` with suffix `dc=p2`
- `oud3` is OUD proxy server which front-ends `OUD1` and `OUD2` above using `ProxyLDAPWorkflowElement` `pwe1` and `pwe2` respectively. `DynamicGroupsWorkflowElement` is configured on top of `ProxyLDAPWorkflowElement` `pwe1` in order to handle the processing of Dynamic groups present in `oud1`.
- `LDAPExt1` and `LDAPExt2` are `LDAPServerExtensions`.

24.9.2 Implementing DynamicGroup Workflow Element Configuration Model

Learn how to implement DynamicGroup Workflow element configuration.

The following sections describe how to implement dynamic group workflow element configuration:

- Setting Up OUD Instances to Configure DynamicGroups Workflow Element
- Configuring Proxy LDAP Workflow Element and DynamicGroups Workflow in First OUD Instance
- Configuring Proxy LDAP Workflow Element in Second OUD Instance

24.9.2.1 Setting up OUD Instances to Configure DynamicGroups Workflow Element

You need to setup OUD proxy and OUD instances to configure dynamic groups.

1. Run `oud-setup` to create an instance `oud1`with baseDN `dc=p1`.

```
./oud-setup --cli --no-prompt --hostname localhost --ldapPort 2389 --rootUserDN
cn="Directory Manager" --rootUserPasswordFile pwd.txt --baseDN dc=p1 --
adminConnectorPort 2444
```

2. Run `oud-setup`to create another instance `oud2` with baseDN `dc=p2`.

```
./oud-setup --cli --no-prompt --hostname localhost --ldapPort 3389 --rootUserDN
cn="Directory Manager" --rootUserPasswordFile pwd.txt --baseDN dc=p2 --
adminConnectorPort 3444
```

3. Run `oud-proxy-setup` to create a proxy server instance `oud3`.

```
./oud-proxy-setup --cli --adminConnectorPort 4444 --ldapPort 4389 --rootUserDN
"cn=Directory Manager" --rootUserPasswordFile  pwd.txt --no-prompt -noPropertiesFile
```

24.9.2.2 Configuring Proxy LDAP Workflow Element and DynamicGroups Workflow Against First OUD Instance

Learn how to configure the Proxy LDAP Workflow Element and dynamic groups Workflow Element against `oud1`.

Run the following command to create LDAP Server extension, a workflow, a dynamic group and associate it with network group.

1. Run `dsconfig create-extension` to create LDAP Server Extension.

```
./dsconfig create-extension
--set enabled:true
--set remote-ldap-server-address:host01.example.com
--set remote-ldap-server-port:2389
--type ldap-server --extension-name LdapExt1
--hostname host01.example.com
--port 4444
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

2. Run `dsconfig create-workflow-element` to create an ldap proxy workflow element that points to `oud1`.

```
./dsconfig create-workflow-element
--set client-cred-mode:use-client-identity
--set enabled:true
--set ldap-server-extension:LdapExt1
--set remote-ldap-server-bind-dn:cn="Directory Manager"
--set remote-ldap-server-bind-password:pwd.txt
--type proxy-ldap --element-name pwe1
--hostname host01.example.com
--port 4444
--portProtocol LDAP
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

3. Run `dsconfig create-workflow-element` with `global-search` parameter set to `true` and `user-search-base` set to `dc=p2`.

```
 ./dsconfig create-workflow-element
--set enabled:true
--set next-workflow-element:pwe1
--set global-search:true
```

```
--set user-search-base:dc=p2
--type dynamic-groups
--element-name dgwe
--hostname host01.example.com
--port 4444
--portProtocol LDAP
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

4. Run `dsconfig create-workflow` to create a generic workflow with name `dgw` and base-dn set to `dc=p1`.

```
./dsconfig create-workflow
--set enabled:true
--set base-dn:dc=p1
--set workflow-element:dgwe
--type generic
--workflow-name dgw
--hostname host01.example.com
--port 4444
--portProtocol LDAP
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

5. Run `set-network-group-prop` to add workflow `dgw`.

```
./dsconfig set-network-group-prop
--group-name network-group
--set workflow:dgw
--hostname host01.example.com
--port 4444
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

6. Run an `ldapsearch` to verify the configuration with `dc=p1`.

```
./ldapsearch -p 4389 -D cn="Directory Manager" -w password -s sub -b dc=p1
"objectclass=*" uid
```

24.9.2.3 Configuring LDAP Proxy Workflow Element Against Second OUD Instance

Learn how to configure Proxy LDAP Workflow Element against `oud2` with `dc=p2`.

1. Run `dsconfig create-extension` to create an LDAP server extension `LDAPext2` that points to `oud2`.

```
./dsconfig create-extension
--set enabled:true
 --set remote-ldap-server-address:localhost
--set remote-ldap-server-port:3389
--type ldap-server
--extension-name LDAPext2
--hostname host01.example.com
--port 4444
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

2. Create the a Proxy LDAP Workflow Element `ProxyWe2` that points to `oud2`.

```
./dsconfig create-workflow-element
--set client-cred-mode:use-client-identity
--set enabled:true
--set ldap-server-extension:LDAPext2
--set remote-ldap-server-bind-dn:cn="Directory Manager"
--set remote-ldap-server-bind-password:pwd.txt
--type proxy-ldap
--element-name ProxyWe2
--hostname host01.example.com
--port 4444
--portProtocol LDAP
--bindDN cn=”Directory Manager”
--bindPasswordFile pwd.txt
--no-prompt
```

3. Create a workflow `p2w` for `dc=p2`.

```
./dsconfig create-workflow
--set enabled:true
--set base-dn:dc=p2
--type generic
--set workflow-element:pwe2
--workflow-name p2w
--hostname host01.example.com
--port 4444
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

4. Run `set-network-group-prop` to add the workflow `p2w`.

```
./dsconfig set-network-group-prop
--group-name network-group
--add workflow:p2w
--hostname localhost
--port 4444
--bindDN cn="Directory Manager"
--bindPasswordFile pwd.txt
--no-prompt
```

5. Run an ldapsearch against OUD proxy server with base as `dc=p2`. Now the results should be drawn from `oud2` as well which is linked to proxy ldap workflow and to the network group.

```
./ldapsearch -p 4389 -D "cn=Directory Manager" -w password -s sub -b dc=p2
"objectclass=*" uid
```

24.9.3 Testing the DynamicGroup Workflow Element Configuration

Learn how to test the dynamic group workflow configuration.

The following sections describe how to test the dynamic groups configuration:

- Testing the DynamicGroups With and Without Expanding memberURL Attribute
- Testing Group Membership

24.9.3.1 Testing DynamicGroups with and without expanding `memberURL` attribute

Learn how to test dynamic groups with and without expanding `memberURL` attribute.

Perform the following steps to check the LDAP entries based on `memberURL` attribute expansion:

1. Assume the following LDAP entry in an LDIF file.

```
dn:cn=admingroup,dc=groups,dc=acme,dc=com
uniqueMember:cn=mark,cn=users,dc=acme,dc=com
memberURL:ldap:///cn=users,dc=acme,dc=com??sub?(|(cn=john)(cn=smith))
objectClass:groupOfUniqueNames
objectClass:groupOfUrls
```

2. The following LDAP search returns the entry as is; without expanding the `memberURL` value. The query matches the dynamic group entry, as it has `cn=mark,cn=users,dc=acme,dc=com` as a static group member via `uniqueMember` attribute.

```
Base DN: dc=groups,dc=acme,dc=com
Scope:sub
Filter:uniqueMember=cn=mark,cn=users,dc=acme,dc=com
```

3. However, if you execute the following LDAP search, the Dynamic Group workflow returns the following LDAP entry with the `cn=john` value in `uniqueMember`.

```
Base DN: dc=groups,dc=acme,dc=com
Scope:sub
Filter:uniquemember=cn=john,cn=users,dc=acme,dc=com

dn:cn=admingroup,dc=groups,dc=acme,dc=com
uniqueMember:cn=mark,cn=users,dc=acme,dc=com
uniqueMember:cn=john,cn=users,dc=acme,dc=com
memberURL:ldap:///cn=users,dc=acme,dc=com??sub?(|(cn=john)(cn=smith))
objectClass:groupOfUniqueNames
objectClass:groupOfUrls
```

   In this query the search filter does not match the dynamic group entry as it does not have `cn=john,cn=users,dc=acme,dc=com` defined as a static group member. But since the query does not return any results, the DynamicGroups workflow element processes all the dynamic groups by expanding their `memberURL` attributes to check if any of them match the search filter. It finds that admingroup entry (after expanding `memberURL`) matches the search filter, so returns it.

> **Note:**
>
> If there is `objectclass=groupOfUniqueNames` and `objectclass=groupOfUrls` in the DynamicGroup, then you have to change `single-structural-objectclass-behavior:accept` parameter using the `--advanced` option as follows:
>
> ```
> ./dsconfig set-global-configuration-prop --set single-structural-objectclass-behavior:accept --advanced
> ```
>
> To configure the `single-structural-objectclass-behavior:accept` parameter, see "`single-structural-objectclass-behavior`" in the *Configuration Reference for Oracle Unified Directory*.

24.9.3.2 Testing Group Membership

The Dynamic Groups workflow element detects a membership test query by detecting the presence of both `cn` and `uniqueMember` filter terms. When present, the Dynamic Groups workflow processes the query differently by recognizing that the client wants to test a membership assertion. The workflow element modifies the results and returns only the single user being tested as the member.

- Run the following `ldapsearch` to test the group membership:

```
./ldapsearch -D bindDN -q -b ou=groups,ou=airius,o=yourcompany.com -s
sub "(&(cn=TestCheck)(uniquemember=cn=Jim Ward,ou=accounting,o=yourcompany.com))"

cn: TestCheck,ou=groups,ou=airius,o=yourcompany.com
memberURL:ldap:///oud=accounting,o=yourcompany.com??sub?(cn=*)
objectClass: groupOfUniqueNames
objectClass: groupOfUrls
cn: TestCheck
uniqueMember: cn=Jim Ward,ou=Accounting,o=YourCompany,com
```

```
cn=Jim WarduniqueMember
```

25

Configuring Proxy, Distribution, and Virtualization Deployments

You can configure a proxy server in various deployment scenarios by using the `dsconfig` command.
The following topics describe how to configure a proxy server by using the `dsconfig` command:

- Configuring a Load Balancing Deployment
- Configuring a Distribution Deployment
- Configuring a Distribution Deployment with Load Balancing
- Configuring a Failover Deployment Between Data Centers
- Configuring a Distribution with Failover Deployment Between Data Centers
- Configuring a Union Workflow Element Deployment with Union Partition

> **Note:**
>
> You can also perform these configurations by using `dsconfig` in interactive mode. For information, see Using `dsconfig` in Interactive Mode.

> **Note:**
>
> To use the virtual directory capabilities described here, you must have a valid `Oracle Directory Service Plus` license.

25.1 Configuring a Load Balancing Deployment

You can create and define different objects when configuring a proxy server for simple load balancing with failover on two LDAP servers.

The following topics describe creating and configuring simple load balancing:

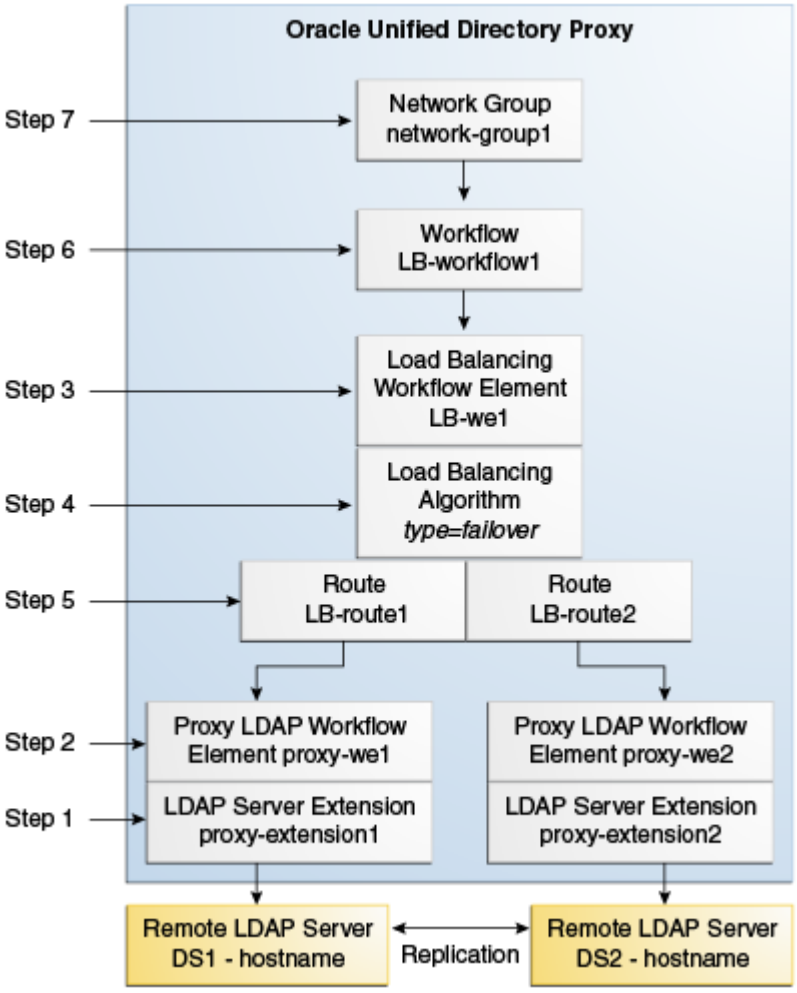
- Creating Objects for Simple Load Balancing
- Configuring a Simple Load Balancing

25.1.1 Creating Objects for Simple Load Balancing

The objects that are to be created when configuring a proxy server for simple load balancing are described in this section.

When configuring a proxy server for simple load balancing, you must create the objects shown in Figure 25-1. You must create these objects in the order indicated.

Figure 25-1 Load Balancing



All of the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`), and the file containing the proxy password (`-j`). You must also indicate the authentication. If you do not specify authentication, and if the client and server are running in the same instance, then Oracle Unified Directory uses the local authentication configuration.

25.1.2 Configuring a Simple Load Balancing

You can configure a simple load balancing deployment by creating a proxy LDAP server extension and a workflow element for each LDAP server extension.

To configure a simple load balancing deployment:

1. Create a proxy LDAP server extension.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

   The LDAP server extension is a link to the remote LDAP server. For this task, you need at least two remote LDAP server instances. Repeat this step, using a *different* LDAP hostname and port for each server.

2. Create a proxy workflow element for each LDAP server extension.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

   The property `client-cred-mode` indicates the type of authentication used between the proxy and remote LDAP server. The client credential mode can be: `use-client-identity` or `use-specific-identity`.

3. Create a load balancing workflow element.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name LB-we1 \
--type load-balancing \
--set enabled:true
```

   You only need one load balancing workflow element to route requests to either of the two remote LDAP servers.

4. Define the load balancing algorithm.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-load-balancing-algorithm \
--element-name LB-we1 \
--type failover
```

   The load balancing algorithm types include `proportional`, `saturation`, `optimal`, `searchfilter` or `failover`. You define the load balancing algorithm properties (weight, threshold, or priority) with the load balancing routes, in the next step.

5. Define the load balancing routes for each proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-load-balancing-route \
--element-name LB-we1 \
```

```
--route-name LB-route1 \
--type failover \
--set workflow-element:proxy-we1 \
--set add-priority:1 \
--set bind-priority:2 \
--set compare-priority:2 \
--set delete-priority:1 \
--set extended-priority:2 \
--set modify-priority:1 \
--set modifydn-priority:1 \
--set search-priority:2
```

When defining the routes, you must specify the same type that you used when defining the load balancing algorithm.

For this task, you need two load balancing routes. Repeat this step, specifying a different priority for each route.

> **Note:**
>
> The properties in this step set the priority for `failover` load balancing. You use different properties for `proportional` or `saturation` load balancing.
>
> For more information on the setting different load balancing types, see Modifying Load Balancing Properties.

6. Create a workflow.

   This workflow associates the load balancing workflow element with the specified base dn.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow \
--workflow-name LB-workflow1 \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:LB-we1
```

7. Create the network group.

   The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-network-group \
--group-name network-group1 \
--set enabled:true \
--set workflow:LB-workflow1 \
--set priority:1
```

25.2 Configuring a Distribution Deployment

You can create and define different objects when configuring a proxy server for a simple distribution deployment that is split over two partitions.

This following topics describe creating and configuring a distribution deployment:

- Creating Objects for Simple Distribution
- Configuring a Simple Distribution Deployment

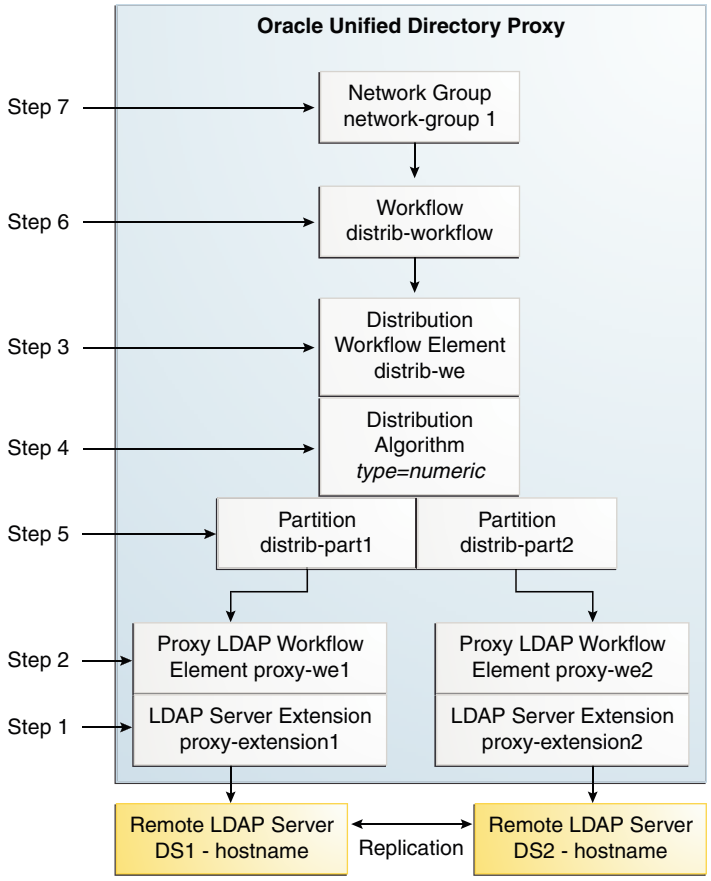
> **Note:**
>
> For information about the supported distribution types, see Overview of Data Distribution Using the Proxy.

25.2.1 Creating Objects for Simple Distribution

You can create objects when configuring a proxy server for simple distribution.

When configuring a proxy server for simple distribution, you must create the objects shown in Figure 25-2. You must create these objects in the order indicated.

Figure 25-2 Configuring Distribution



All of the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`) and the proxy password you want to configure (`-w`). You must also indicate the authentication. If you do not specify authentication, and if the client and server are running in the same instance, then Oracle Unified Directory uses the local authentication configuration.

25.2.2 Configuring a Simple Distribution Deployment

You can configure a simple distribution deployment by creating a proxy LDAP server extension.

To configure a simple distribution deployment:

1. Create a proxy LDAP server extension.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

   The LDAP server extension is a link to the remote LDAP server. For this task, you need two remote LDAP server instances. Repeat this step, using a *different* LDAP hostname and port for each server.

2. Create a proxy workflow element for each LDAP server extension.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

   You need at least two remote LDAP servers for a distribution architecture. Repeat this step for each server. Use the same LDAP server extension names that you created in step 1.

   The property `client-cred-mode` indicates the type of authentication used between the proxy and remote LDAP server. The client credential mode can be: `use-client-identity` or `use-specific-identity`.

3. Set up distribution by creating a distribution workflow element.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name distrib-we \
--type distribution \
--set base-dn:dc=example,dc=com \
--set enabled:true
```

4. Set the distribution algorithm.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-distribution-algorithm \
--element-name distrib-we \
--type numeric \
--set distribution-attribute:uid
```

   The distribution algorithm types include `capacity`, `numeric`, `lexico`, or `dnpattern`. You define the algorithm properties when you create the distribution partitions, in the next step.

5. Define the distribution partitions.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-distribution-partition \
--element-name distrib-we \
--partition-name distrib-part1\
```

```
--type numeric \
--set lower-bound:0 \
--set upper-bound:1000 \
--set partition-id:1 \
--set workflow-element:proxy-we1
```

For this task, you must create two partitions. You must use unique partition IDs and partition names for each workflow element. When defining the partitions, you must specify the same type that you used when defining the distribution algorithm.

> **Note:**
>
> The upper boundary is *exclusive*, which means if you specify `upper-bound:1000` as the upper boundary, then the partition only includes values from 0 to 999, inclusive.

- If you created a `capacity` distribution algorithm, then you must create a global index.
- If you created a `lexico`, `numeric`, or `dnpattern` distribution algorithm, then creating a global index is optional.

To create a global index:

a. Create a global index catalog.

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-catalog \
--catalogName gi-catalog
```

b. Add a global index that indexes the dn attribute to the catalog.

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
add-index \
--catalogName gi-catalog \
--attributeName dn
```

c. Associate the global index catalog to the distribution.

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
associate \
--catalogName gi-catalog \
--distributionWorkflowElement distrib-we
```

6. Create a workflow.

   This workflow associates the distribution workflow element with the distribution partition.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow \
--workflow-name distrib-workflow \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:distrib-we
```

7. Create the network group.

   The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-network-group \
--group-name network-group1 \
--set enabled:true \
```

```
--set workflow:distrib-workflow \
--set priority:1
```

25.3 Configuring a Distribution Deployment with Load Balancing

You can create and define different objects when configuring a proxy server for a distribution deployment with load balancing.

The following topics describe creating and configuring distribution with load balancing deployment:

- Creating Objects for Distribution with Load Balancing
- Configuring a Distribution with Load Balancing Deployment

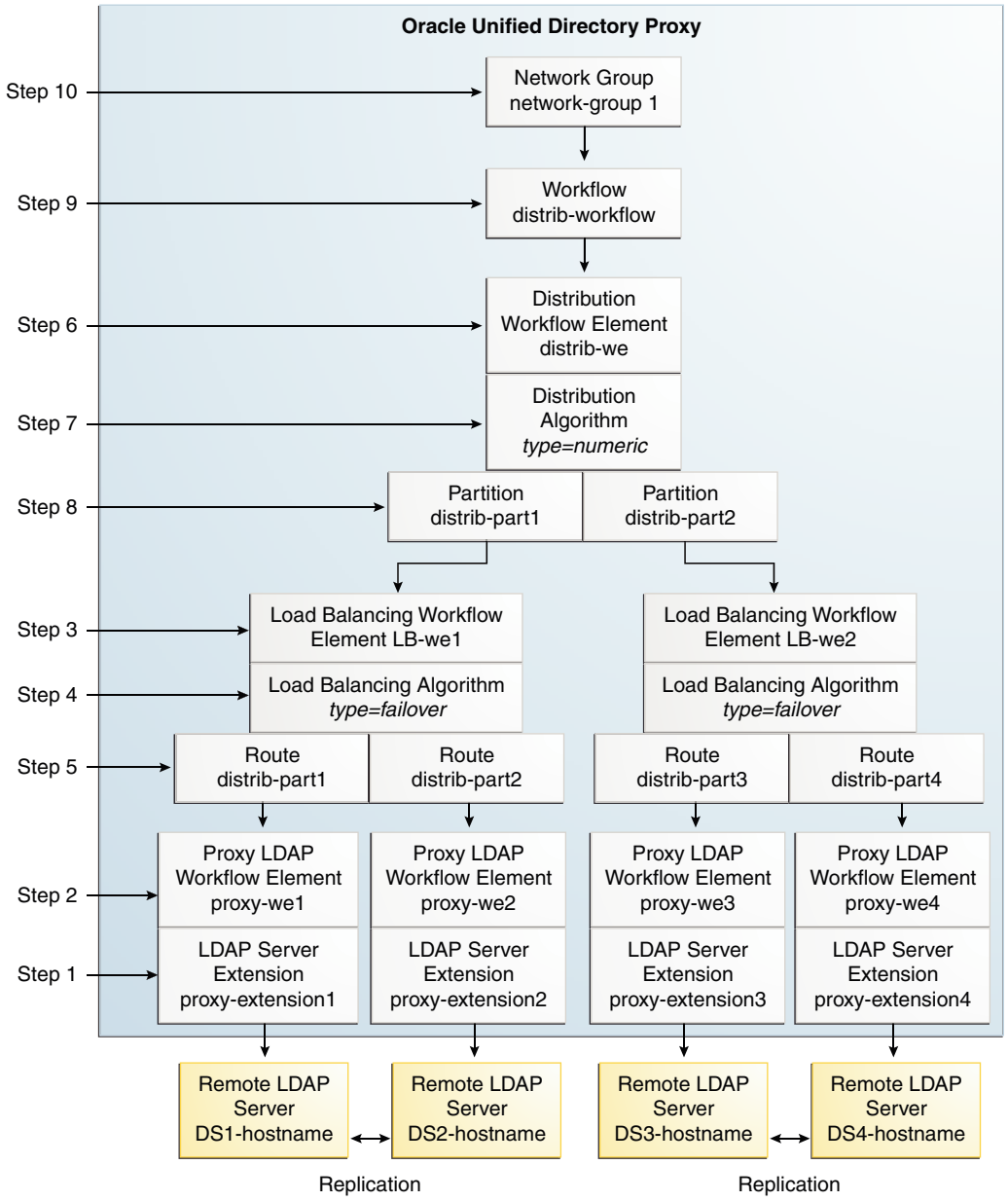
> **Note:**
>
> Although you can add a global index to any distribution deployment, this example does not include instructions for adding a global index.
>
> For information about creating a global index, see Configuring Global Indexes Using the Command Line.

25.3.1 Creating Objects for Distribution with Load Balancing

You can create objects when configuring a proxy server for distribution and load balancing.

When configuring a proxy server for distribution and load balancing, you must create the objects shown in Figure 25-3. You must create these objects in the order indicated.

Figure 25-3 Configuring Distribution and Load Balancing



This example illustrates a deployment with distribution over two partitions, with each partition load balanced onto two replicated LDAP servers. The example uses a `numeric` distribution algorithm to partition the data.

All of the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`), and the file containing the proxy password (`-j`). You must also indicate the authentication. If you do not specify authentication, and if the client and server are running in the same instance, then Oracle Unified Directory uses the local authentication configuration.

25.3.2 Configuring a Distribution with Load Balancing Deployment

You can use the procedure to configure a distribution with load balancing deployment.

To configure a distribution and load balancing deployment:

1. Create the proxy LDAP server extensions.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

   The LDAP server extension is a link to the remote LDAP server. For this task, you need four remote LDAP server instances. Repeat this step for each remote LDAP server, using a different LDAP hostname and port for each server.

2. Create a proxy workflow element for each LDAP server extension.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

   For this task, you need four remote LDAP server instances. Repeat this step for each remote server, using the same LDAP server extension names as those created in step 1.

   The property `client-cred-mode` indicates the type of authentication used between the proxy and remote LDAP server. The client credential mode can be: `use-client-identity` or `use-specific-identity`.

3. Create a load balancing workflow element.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name LB-we1 \
--type load-balancing \
--set enabled:true
```

   You only need one load balancing workflow element to route requests to either of the two remote LDAP servers. You must create two load balancing workflow elements because you are using two load balancers.

4. Define the load balancing algorithm.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-load-balancing-algorithm \
--element-name LB-we1 \
--type failover
```

   The load balancing algorithm types include `proportional`, `optimal`, `saturation`, `searchfilter`, or `failover`. You define the load balancing algorithm properties (weight, threshold, or priority) with the load balancing routes, in the next step. For this task, you need two load balancing algorithms.

5. Define the load balancing routes for each proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-load-balancing-route \
--element-name LB-we1 \
--route-name LB-route1 \
--type failover \
--set workflow-element:proxy-we1 \
--set add-priority:1 \
--set bind-priority:1 \
--set compare-priority:1 \
--set delete-priority:1 \
--set extended-priority:1 \
--set modify-priority:1 \
--set modifydn-priority:1 \
--set search-priority:1
```

For this task, you need four load balancing routes. Set two routes per load balancing workflow element (created in step 4). For example, set one route with priority 1 for all operations and set the other route with priority 2 for all operations.

> **Note:**
>
> The properties in this step set the priority for `failover` load balancing. You use different properties for `proportional` or `saturation` load balancing.
>
> For more information on the setting different load balancing types, see Modifying Load Balancing Properties.

6. Set up distribution by creating a distribution workflow element.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow-element \
--element-name distrib-we \
--type distribution \
--set base-dn:dc=example,dc=com \
--set enabled:true
```

For this task, you need only one distribution workflow element that points to the distribution algorithm.

7. Set the distribution algorithm.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-distribution-algorithm \
--element-name distrib-we \
--type numeric \
--set distribution-attribute:uid
```

The distribution algorithm types include `capacity`, `numeric`, `lexico`, or `dnpattern`. You define the boundaries when you create the distribution partitions, in the next step.

8. Define the distribution partitions.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-distribution-partition \
--element-name distrib-we \
--partition-name distrib-part1\
--type numeric \
--set lower-bound:0 \
--set upper-bound:1000 \
```

```
--set partition-id:1 \
--set workflow-element:LB-we1
```

For this task, you must create two partitions. You must use unique partition IDs and partition names for each workflow element and that each partition uses a different load balancing workflow element. When defining the partitions, you must specify the same type that you used when defining the distribution algorithm.

> **Note:**
>
> The upper boundary is *exclusive*, which means if you specify `upper-bound:1000` as the upper boundary, then the partition only includes values from 0 to 999, inclusive.

- If you created a `capacity` distribution algorithm, then you must create a global index.
- If you created a `lexico`, `numeric`, or `dnpattern` distribution algorithm, then creating a global index is optional.

To create a global index.

a. Create a global index catalog:

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-catalog \
--catalogName gi-catalog
```

b. Add a global index which indexes the dn attribute to the catalog.

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
add-index \
--catalogName gi-catalog \
--attributeName dn
```

c. Associate the global index catalog to the distribution.

```
$ gicadm -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
associate \
--catalogName gi-catalog \
--distributionWorkflowElement distrib-we
```

9. Create a workflow.

   This workflow associates the distribution workflow element with the base DN.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-workflow \
--workflow-name workflow \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:distrib-we
```

10. Create the network group.

    The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file \
create-network-group \
--group-name network-group1 \
--set enabled:true \
--set workflow:workflow \
--set priority:1
```

25.4 Configuring a Failover Deployment Between Data Centers

To set up a failover deployment between two data centers, you need to deploy two levels of load balancers within the proxy.

To set up a failover deployment between two data centers, see Configuration 3: Failover Between Data Centers.

To configure a failover deployment between two data centers, use the following commands:

```
#Create a proxy LDAP extension for each remote LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension1 \
  --set enabled:true \
  --set remote-ldap-server-address:DS1_hostname \
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension2 \
  --set enabled:true \
  --set remote-ldap-server-address:DS2_hostname \
  --set remote-ldap-server-port:3289

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension3 \
  --set enabled:true \
  --set remote-ldap-server-address:DS3_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension4 \
  --set enabled:true \
  --set remote-ldap-server-address:DS4_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we1 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we2 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension2
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we3 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension3

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we4 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension4

# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name LB-we1 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name LB-we2 \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we1 \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we2 \
  --type proportional

# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we1 \
  --route-name LB-route1 \
  --type proportional \
  --set workflow-element:proxy-we1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we1 \
  --route-name LB-route2 \
  --type proportional \
  --set workflow-element:proxy-we2

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we2 \
  --route-name LB-route3 \
  --type proportional \
  --set workflow-element:proxy-we3
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we2 \
  --route-name LB-route4 \
  --type proportional \
  --set workflow-element:proxy-we4

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name FO-we \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name FO-we \
  --type failover

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we \
  --route-name FO-route1 \
  --type failover \
  --set workflow-element:LB-we1 \
  --set add-priority:1 \
--set bind-priority:1 \
--set compare-priority:1 \
--set delete-priority:1 \
--set extended-priority:1 \
--set modify-priority:1 \
--set modifydn-priority:1 \
--set search-priority:1 \

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we \
  --route-name FO-route2 \
  --type failover \
  --set workflow-element:LB-we2 \
  --set add-priority:2 \
--set bind-priority:2 \
--set compare-priority:2 \
--set delete-priority:2 \
--set extended-priority:2 \
--set modify-priority:2 \
--set modifydn-priority:2 \
--set search-priority:2 \

# Create workflow
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow \
  --workflow-name FO-workflow \
  --set enabled:true \
  --set base-dn:dc=example,dc=com \
  --set workflow-element:FO-we

# Create network group
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-network-group \
  --group-name network-group1 \
  --set enabled:true \
```

```
  --set workflow:FO-workflow \
  --set priority:1
```

25.5 Configuring a Distribution with Failover Deployment Between Data Centers

To set up a failover deployment between two data centers, you can configure a topology that includes distribution with failover load balancing.

To set up a failover deployment between two data centers, see Configuration 5: Distribution with Failover Between Data Centers.

To configure distribution with failover between two data centers, use the following commands:

```
#Create the first failover route
#Create a proxy LDAP extension for each remote LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-1a \
  --set enabled:true \
  --set remote-ldap-server-address:DS1a_hostname \
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-2a \
  --set enabled:true \
  --set remote-ldap-server-address:DS2a_hostname \
  --set remote-ldap-server-port:3289

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-1b \
  --set enabled:true \
  --set remote-ldap-server-address:DS1b_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-2b \
  --set enabled:true \
  --set remote-ldap-server-address:DS2b_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-1a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-1a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-2a \
```

```
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-2a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-1b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-1b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-2b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-2b

# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name LB-we-1a \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name LB-we-1b \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-1a \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-1b \
  --type proportional

# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-1a \
  --route-name LB-route-1a \
  --type proportional \
  --set workflow-element:proxy-we-1a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-1a \
  --route-name LB-route-2a \
  --type proportional \
  --set workflow-element:proxy-we-2a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
```

```
  --element-name LB-we-1b \
  --route-name LB-route-1b \
  --type proportional \
  --set workflow-element:proxy-we-1b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-1b \
  --route-name LB-route-2b \
  --type proportional \
  --set workflow-element:proxy-we-2b

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name FO-we1 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name FO-we1 \
  --type failover

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we1 \
  --route-name FO-route-1a \
  --type failover \
  --set workflow-element:LB-we-1a \
  --set add-priority:1 \
  --set bind-priority:1 \
  --set compare-priority:1 \
  --set delete-priority:1 \
  --set extended-priority:1 \
  --set modify-priority:1 \
  --set modifydn-priority:1 \
  --set search-priority:1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we1 \
  --route-name FO-route-1b \
  --type failover \
  --set workflow-element:LB-we-1b \
  --set add-priority:2 \
  --set bind-priority:2 \
  --set compare-priority:2 \
  --set delete-priority:2 \
  --set extended-priority:2 \
  --set modify-priority:2 \
  --set modifydn-priority:2 \
  --set search-priority:2

#Create the second failover route
#Create a proxy LDAP extension for each remote LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-3a \
  --set enabled:true \
  --set remote-ldap-server-address:DS3a_hostname \
```

```
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-4a \
  --set enabled:true \
  --set remote-ldap-server-address:DS4a_hostname \
  --set remote-ldap-server-port:3289

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-3b \
  --set enabled:true \
  --set remote-ldap-server-address:DS3b_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-4b \
  --set enabled:true \
  --set remote-ldap-server-address:DS4b_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-3a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-3a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-4a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-4a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-3b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-3b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name proxy-we-4b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-4b

# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
```

```
  --element-name LB-we-2a \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name LB-we-2b \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-2a \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-2b \
  --type proportional

# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-2a \
  --route-name LB-route-3a \
  --type proportional \
  --set workflow-element:proxy-we-3a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-2a \
  --route-name LB-route-4a \
  --type proportional \
  --set workflow-element:proxy-we-4a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-2b \
  --route-name LB-route-3b \
  --type proportional \
  --set workflow-element:proxy-we-3b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name LB-we-2b \
  --route-name LB-route-4b \
  --type proportional \
  --set workflow-element:proxy-we-4b

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name FO-we2 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-algorithm \
  --element-name FO-we2 \
  --type failover
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we2 \
  --route-name FO-route-2a \
  --type failover \
  --set workflow-element:LB-we-2a \
  --set add-priority:1 \
  --set bind-priority:1 \
  --set compare-priority:1 \
  --set delete-priority:1 \
  --set extended-priority:1 \
  --set modify-priority:1 \
  --set modifydn-priority:1 \
  --set search-priority:1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-load-balancing-route \
  --element-name FO-we2 \
  --route-name FO-route-2b \
  --type failover \
  --set workflow-element:LB-we-2b \
  --set add-priority:2 \
  --set bind-priority:2 \
  --set compare-priority:2 \
  --set delete-priority:2 \
  --set extended-priority:2 \
  --set modify-priority:2 \
  --set modifydn-priority:2 \
  --set search-priority:2

# Create distribution to the two failover routes
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow-element \
  --element-name distrib-we \
  --type distribution \
  --set base-dn:dc=example,dc=com \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-distribution-algorithm \
  --element-name distrib-we \
  --type numeric \
  --set distribution-attribute:uid

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-part1\
  --type numeric \
  --set lower-bound:0 \
  --set upper-bound:1000 \
  --set partition-id:1 \
  --set workflow-element:FO-we1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-part2\
  --type numeric \
  --set lower-bound:1000 \
  --set upper-bound:2000 \
  --set partition-id:2 \
```

```
  --set workflow-element:FO-we2


# Create workflow
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-workflow \
  --workflow-name Distrib-workflow \
  --set enabled:true \
  --set base-dn:dc=example,dc=com \
  --set workflow-element:distrib-we

# Create network group
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -j pwd-file -X -n \
create-network-group \
  --group-name network-group1 \
  --set enabled:true \
  --set workflow:Distrib-workflow \
  --set priority:1
```

25.6 Configuring a Union Workflow Element Deployment with Union Partition

You can create and configure a Union workflow element to aggregate several DITs into a virtual unified DIT by using `dsconfig` command.

- Setting Up OUD Instances to Implement Union Workflow Element Configuration
- Setting Up OUD Proxy Server To Implement Union Workflow Element Configuration
- Configuring OUD Proxy Server to Implement Union Workflow Element Configuration
- Creating Union Workflow Element
- Configuring Union Workflow Element
- Configuring Union Partition
- Validating Union Workflow Element Configuration

25.6.1 Setting Up OUD Instances to Implement Union Workflow Element Configuration

You must create and configure the OUD (`oud1` and `oud2`) instances to implement the Union workflow element configuration.

Perform the following steps:

1. Run the `oud-setup` command to create the `oud1` instance as follows:

   ```
   ./oud-setup --cli --baseDN "o=company1" --addBaseEntry --
   adminConnectorPort 1444 --ldapPort 1389 \
   --rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt --no-
   prompt --noPropertiesFile
   ```

2. Run the `oud-setup` command to create the `oud2` instance as follows:

   ```
   ./oud-setup --cli --baseDN "o=company2" --addBaseEntry --
   adminConnectorPort 2444 --ldapPort 2389 \
   ```

```
--rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt --no-
prompt --noPropertiesFile
```

25.6.2 Setting Up OUD Proxy Server to Implement Union Workflow Element Configuration

This topic illustrates how to set up the OUD proxy server to implement Union workflow element configuration.

Run the `oud-setup` command to create a proxy server instance (`oud3`):

```
./oud-proxy-setup --cli --ldapPort 3389 --adminConnectorPort 3444 --
rootUserDN "cn=Directory Manager" --rootUserPasswordFile pwd.txt
```

25.6.3 Configuring OUD Proxy Server to Implement Union Workflow Element Configuration

To connect to a remote LDAP directory server, the Oracle Unified Directory proxy needs LDAP server extension and LDAP proxy workflow element.

LDAP Server extensions are the properties required to connect from OUD Proxy `oud3` to the remote LDAP servers (`oud1` and `oud2`). You create LDAP Server extensions for `oud1` and `oud2`. You will use these extensions in the workflow configuration later.

In addition, you need to create proxy workflow elements for `OUD1` and `OUD2`. These specify connection details and credentials to the remote LDAP servers.

1. Create an LDAP Server Extension (`LDAPServerExtension1`) and that points to `oud1`.

```
./dsconfig create-extension \
        --set enabled:true \
        --set remote-ldap-server-address:host01.example.com \
        --set remote-ldap-server-port:1389 \
        --type ldap-server \
        --extension-name LDAPServerExtension1 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

2. Create an LDAP Proxy workflow element (`ProxyLDAPWorkFlowElement1`) that points to `oud1`.

```
./dsconfig create-workflow-element \
        --set client-cred-mode:use-client-identity \
        --set enabled:true \
                --set remote-ldap-server-bind-dn:"cn=directory manager" \
                --remote-ldap-server-bind-password:file://pwd-file \
        --set ldap-server-extension:LDAPServerExtension1 \
                --set remote-root-dn:"cn=directory manager"
                --set remote-root-password:********
```

```
        --type proxy-ldap \
        --element-name ProxyLDAPWorkFlowElement1 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

3. Create an LDAP Server Extension (`LDAPServerExtension2`) that points to `oud2`.

```
./dsconfig create-extension \
        --set enabled:true \
        --set remote-ldap-server-address:host01.example.com \
        --set remote-ldap-server-port:2389 \
        --type ldap-server \
        --extension-name LDAPServerExtension2 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

4. Create an LDAP Proxy workflow element (`ProxyLDAPWorkFlowElement2`) that points to `oud2`.

```
./dsconfig create-workflow-element \
        --set client-cred-mode:use-client-identity \
        --set enabled:true \
                --set remote-ldap-server-bind-dn:"cn=directory manager" \
                --remote-ldap-server-bind-password:file://pwd-file \
        --set ldap-server-extension:LDAPServerExtension2 \
                --set remote-root-dn:"cn=directory manager"
                --set remote-root-password:********
        --type proxy-ldap \
        --element-name ProxyLDAPWorkFlowElement2 \
        --hostname host01.example.com \
        --port 3444 \
        --trustAll \
        --bindDN cn=Directory\ Manager \
        --bindPasswordFile pwd.txt \
        --no-prompt
```

5. Run the `dsconfig` command to view the server extensions.

```
./dsconfig -h host01.example.com -p 3444 -D "cn=Directory Manager" --
bindPasswordFile pwd.txt -X -n list-extensions
```

6. Run the `dsconfig` command to view the proxy LDAP workflow elements.

```
./dsconfig -h host01.example.com -p 3444 -D "cn=Directory Manager" --
bindPasswordFile pwd.txt -X -n list-workflow-elements
```

25.6.4 Creating Union Workflow Element

You can create a Union workflow using `dsconfig` command.

To create a new Union Workflow element, run the `dsconfig` command:

```
$ ./dsconfig create-workflow-element \
  --set base-dn:o=company1 \
  --set enabled:true
  --type union \
  --element-name union-we \
  --set enabled:true
    --hostname host01.example.com \
    --port 3444
    --bindDN "cn=Directory Manager" \
    --bindPasswordFile pwd.txt \
```

You can also edit the properties for the Union workflow element using the `dsconfig` command-line interactive mode.

See the following example to edit propertied using the `dsconfig` command in interactive mode.

```
Oracle Unified Directory Configuration Console Main Menu
What do you want to configure?

    1)  General Configuration                       7)  Virtualization
    2)  Authentication and authorization            8)  Load Balancing
    3)  Schema                                      9)  Distribution
    4)  Replication                                 10) Integration
    5)  Local Data Source                           11) HTTP
    6)  Remote Data Source
    q)  quit

Enter choice: 9

Distribution Management Menu

What would you like to do?

    1)  Distribution Workflow      3)  Global Index Catalogs Shared Cache
        Element                        Extension
    2)  Global Index Catalogs      4)  Union Workflow Element
        Extension

    b)  back
    q)  quit

Enter choice [b]: 4

Union Workflow Element management menu

What would you like to do?

    1)  List existing Union Workflow Elements
    2)  Create a new Union Workflow Element
    3)  View and edit an existing Union Workflow Element
    4)  Delete an existing Union Workflow Element
    5)  >>>> Union Partition management menu

    b)  back
    q)  quit
```

```
Enter choice [b]: 3

>>>> There is only one Union Workflow Element: "union-we". Are you sure that
this is the correct one? (yes / no) [yes]:

>>>> Configure the properties of the Union Workflow Element

         Property                         Value(s)
         ---------------------------------------------------------------------
    1)   auto-tune-search-option          false
    2)   base-dn                          "dc=example,dc=com"
    3)   bind-option                      bind-first-success
    4)   cache-size                       10000
    5)   enabled                          true
    6)   search-first-match               true
    7)   suppress-entry-duplicates        false

    ?)   help
    f)   finish - apply any changes to the Union Workflow Element
    q)   quit

Enter choice [f]: ?

Component name: Union Workflow Element

The Union workflow element is used to aggregate several DITs into
a virtual unified DIT.

Option Types:

 r -- Property value(s) are readable
 w -- Property value(s) are writable
 m -- The property is mandatory
 s -- The property is single-valued
 a -- Administrative action is required for changes to take effect

Property                      Options  Syntax
-------------------------------------------------
auto-tune-search-option       rw-s-    BOOLEAN
base-dn                       r-ms-    DN
bind-option                   rw-s-    OPTION
cache-size                    rw-s-    INTEGER
enabled                       rwms-    BOOLEAN
search-first-match            rw-s-    BOOLEAN
suppress-entry-duplicates     rw-s-    BOOLEAN
-------------------------------------------------
```

For more information on the configuration properties of the Union workflow element, see Configuration Parameters for Union Workflow Element.

25.6.5 Configuring Union Workflow Element

You can configure the Union workflow element using `dsconfig` command.

1. Create a workflow. This workflow associates the Union workflow element with the base DN.

```
./dsconfig create-workflow \
          --set base-dn:o=company1 \
          --set enabled:true
```

```
--set workflow-element:union-we
--type generic \
--workflow-name distrib-workflow \
--hostname host01.example.com \
--port 3444 \
--portProtocol LDAP \
--trustAll \
--bindDN cn=Directory\ Manager \
--bindPasswordFile pwd.txt \
--no-prompt
```

2. Attach the Union workflow (`distrib-workflow`) to the network group.

```
./dsconfig set-network-group-prop \
          --group-name network-group \
          --set workflow:distrib-workflow \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

25.6.6 Configuring Union Partition

You can create a Union workflow element using `dsconfig` command.

To create a new Union Partition do the following:

1. Create the Union Partition for `oud1` by running the `dsconfig create-workflow-element --type generic` command:

```
./dsconfig create-union-partition  \
          --element-name union-we \
          --set workflow-element:ProxyLDAPWorkFlowElement1
          --type generic \
          --partition-name union-part1\
          --set priority:1
          --set source-base-dn:ou=people,ou=apac,o=comapny1 \
          --set relative-base-dn:ou=people \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

For more information on the configuration properties of the Union workflow element, see Configuration Parameters for Union Partition.

2. Create the Union Partition for `oud2` by running the `dsconfig create-workflow-element --type generic` command:

```
./dsconfig create-union-partition  \
          --element-name union-we \
          --set workflow-element:ProxyLDAPWorkFlowElement2
          --type generic \
          --partition-name union-part2\
          --set priority:2
          --set source-base-dn:ou=people,ou=apac,o=company2 \
          --set relative-base-dn:ou=people \
          --hostname host01.example.com \
          --port 3444 \
          --portProtocol LDAP \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

For more information on the configuration properties of the Union workflow element, see Configuration Parameters for Union Partition.

25.6.7 Validating Union Workflow Element Configuration

Learn to validate the Union workflow element configuration.

1. Run the `ldapmodify` command to populate the following data for the `union-part1` union partition:

```
dn: o=company1
objectclass: top
objectclass: organization
o: company1

dn: ou=apac,o=company1
objectclass: top
objectclass: organizationalunit
ou: apac

dn: ou=people,ou=apac,o=company1
objectclass: top
objectclass: organizationalunit
ou: people

dn: uid=user.10,ou=people,ou=apac,o=company1
objectclass: top
objectclass: inetOrgPerson
cn: 10
sn: snof10
uid: user.10
userPassword: 10
```

2. Run the `ldapmodify` command to populate the `union-part2` union partition:

```
dn: o=company2
objectclass: top
objectclass: organization
o: company2

dn: ou=apac,o=company2
objectclass: top
objectclass: organizationalunit
ou: apac

dn: ou=people,ou=apac,o=company2
objectclass: top
objectclass: organizationalunit
ou: people

dn: uid=user.20,ou=people,ou=apac,o=company2
objectclass: top
objectclass: inetOrgPerson
cn: 20
sn: snof20
uid: user.20
userPassword: 20
```

3. Run the `ldapsearch` for the oud-proxy setup to return all the entries from the partitions:

```
./ldapsearch -h localhost -p 3444 -D "cn=Directory Manager" -j pwd.txt -b
"o=company1" -s sub "(objectclass=*)"
```

In response you will get unified data from different partitions. However, the data is present under different base DNs in the partitions.

For the Union workflow element (`union-we`), if you have set the `auto-tune-search-option` and `search-first-match` properties to `true` then the `ldapsearch` will fetch users from both the partitions (`union-part1` and `union-part2`).

Part V

Advanced Administration: Security, Access Control, and Password Policies

All aspects of a deployment that relate to securing the servers themselves or securing the data that is stored in the directory.

This part includes the following chapters:

26

Configuring Security Between Clients and Servers

You can have several mechanisms to secure traffic between the client and the server. The following topics describe the configuring secure traffic between the client and the server:

- Getting SSL Up and Running Quickly
- Configuring Key Manager Providers
- Configuring Trust Manager Providers
- Configuring Certificate Mappers
- Configuring SSL and StartTLS for LDAP and JMX
- Specifying SSL protocol and Cipher Suites in Crypto Manager for Replication
- Overriding System Default Protocols and Cipher Suites for TLS Communication
- Using SASL Authentication
- Configuring SASL Authentication
- Configuring Kerberos and the Oracle Unified Directory Server for GSSAPI SASL Authentication
- Testing SSL, StartTLS, and SASL Authentication With `ldapsearch`
- Debugging SSL Using OpenSSL `s_client` Test Utility
- Debugging SSL or TLS Using Java Debug Information
- Controlling Connection Access Using Allowed and Denied Rules
- Configuring Unlimited Strength Cryptography
- Configuring SSL Protocol and Cipher Suites Using OUDSM

For information about securing access to directory data, see Controlling Access To Data.

For information about configuring security between the proxy and the directory server or data source, see Configuring Security Between the Proxy and the Data Source.

26.1 Getting SSL Up and Running Quickly

Oracle Unified Directory provides several options for configuring and using SSL and StartTLS. The numerous possibilities for configuration might be daunting for those who are unfamiliar with the technology or who just want to get up and running as quickly as possible for testing purposes.

This section provides a list of the steps that must be performed to allow Oracle Unified Directory to accept SSL-based connections using a self-signed certificate.

The procedures in this section assume a knowledge of truststores and keystores.

- For detailed information about keystores, see Configuring Key Manager Providers.
- For detailed information about truststores, see Configuring Trust Manager Providers.

> **Note:**
>
> Using a self-signed certificate is not recommended for production purposes. To install a certificate for production purposes, follow the instructions in Configuring Key Manager Providers.

26.1.1 Setting Up SSL Using an Existing Private Key and Certificate

If you already have a security certificate that was generated with the `openssl` command-line tool, you must create keystores and update the certificate before you can use it with Oracle Unified Directory.

In the following example, these certificate files already exist:

- ca.crt

  Certificate authority public key (certificate)

- mycert.key

  The private key of the previously generated certificate

- mycert.crt

  The public key of the previously generated certificate

To update the existing security certificate:

1. Create a PKCS12 keystore containing both public and private keys.

   In this example, `keystore.p12` is the PKCS12 keystore you are creating:

   ```
   $ openssl pkcs12 -export -out keystore.p12 -inkey
   mycert.key -in mycert.crt -chain -CAfile ca.crt
   -password file:<FILE CONTAINING THE PASSWORD FOR THE PKCS12 KEYSTORE>
   ```

   You can use the generated PKCS12 keystore as described in Using the PKCS #12 Key Manager Provider.

   Or you can complete step 2 and step 3 to import the certificate into a JKS keystore and update the certificate alias.

2. Create a JKS keystore by importing the certificate from the PKCS12 keystore.

   In this example, `keystore.jks` is the JKS keystore you are creating. You must specify `1` as an alias. The alias is required in the step 3.

   If you want to update the alias of the certificate but continue to store the certificate in a PKCS12 keystore, add the argument `-storetype PKCS12` when invoking the following `keytool` command:

   ```
   $ keytool -importkeystore -deststorepass <PASSWORD OF THE JKS KEYSTORE>
    -destkeypass <PASSWORD OF THE JKS KEY> -destkeystore keystore.jks
   -srckeystore keystore.p12 -srcstoretype PKCS12 -srcstorepass
   <PASSWORD OF THE PKCS12 KEYSTORE> -alias 1
   ```

3. Update the alias of the certificate from `1` to `my-server-cert`.

   If you want to update the alias of the certificate but continue to store the certificate in a PKCS12 keystore, add the argument `-storetype PKCS12` when invoking the following `keytool` command:

```
$ keytool -changealias -keystore keystore.jks -alias 1 -destalias my-server-cert -
storepass <PASSWORD OF THE JKS KEYSTORE>
```

Now you can use the JKS keystore `keystore.jks` and the certificate it contains to configure the key manager provider. See Configuring the JKS Key Manager Provider.

26.1.2 Accepting SSL-Based Connections Using a Self-Signed Certificate

This step is required *only* if the SSL and StartTLS settings were not specified during installation, or if you want to change those settings.

This procedure assumes the following:

- Oracle Unified Directory is installed on the system on which you are working.
- The Java `keytool` utility is in your path. If it is not, either add it to your path or provide the complete path to it when invoking the commands. The `keytool` utility is provided with the Java Runtime Environment (JRE).
- The administration connector is listening on the default port (4444) and the `dsconfig` command is accessing the server running on the local host. If this is not the case, the `--port` and `--hostname` options must be specified.

1. Generate a private key for the certificate, using the `keytool` command with the `-genkeypair` option.

   For example:

   ```
   $ keytool -genkeypair -alias server-cert -keyalg rsa \
     -dname "CN=myhost.example.com,O=Example Company,C=US" \
     -keystore config/keystore -storetype JKS
   ```

   - `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. The default name used by the server is `server-cert`.
   - `-keyalg` *algorithm*. Specifies the algorithm that should be used to generate the private key. This should almost always be `rsa`.
   - `-dname` *subject*. Specifies the subject to use for the certificate.

     Change the value of the `-dname` argument so that it is suitable for your environment:

     The value of the `CN` attribute should be the fully-qualified name of the system on which the certificate is being installed.

     The value of the `O` attribute should be the name of your company or organization.

     The value of the `C` attribute should be the two-character abbreviation for your country.
   - `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it does not already exist. The default keystore path used by the server is `config/keystore`.
   - `-keypass` *password*. Specifies the password that should be used to protect the private key in the keystore. If the password is not provided, you will be prompted for it.
   - `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If the password is not provided, you will be prompted for it.
   - `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, for example, the value should always be `JKS`.

   You are prompted for a password to protect the contents of the keystore and for a password to protect the private key.

2. Generate a self-signed certificate for the key.

   For example:

   ```
   $ keytool -selfcert -alias server-cert -validity 1825 \
      -keystore config/keystore -storetype JKS
   ```

   - `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkeypair` option.
   - `-validity` *days*. Specifies the length of time in days that the certificate should be valid. The default validity is 90 days.
   - `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it does not already exist.
   - `-keypass` *password*. Specifies the password that should be used to protect the private key in the keystore. If this is not provided, then you will be interactively prompted for it.
   - `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If this is not provided, then you will be interactively prompted for it.
   - `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

   When you are prompted for the keystore password and private key password, enter the same passwords that you provided in the previous step.

3. Export the public key for the certificate that you created.

   For example:

   ```
   $ keytool -exportcert -alias server-cert -file config/server-cert.txt -rfc \
      -keystore config/keystore -storetype JKS
   ```

4. Create a new trust store and import the server certificate into that trust store.

   For example:

   ```
   $ keytool -importcert -alias server-cert -file config/server-cert.txt \
     -keystore config/truststore -storetype JKS
   ```

5. Use the `dsconfig` command to enable the key manager provider, trust manager provider, and connection handler.

   For example:

   ```
   $ dsconfig -D "cn=directory manager" -j pwd-file -X -n
   set-key-manager-provider-prop --provider-name JKS --set enabled:true
   --set key-store-pin:KEYSTORE_PASSWORD
   $ dsconfig -D "cn=directory manager" -j pwd-file -X -n
   set-trust-manager-provider-prop --provider-name JKS --set enabled:true
   --set trust-store-pin:TRUSTSTORE_PASSWORD
   $ dsconfig -D "cn=directory manager" -j pwd-file -X -n
     set-connection-handler-prop --handler-name "LDAPS Connection Handler"
     --set trust-manager-provider:JKS --set key-manager-provider:JKS
     --set listen-port:1636 --set enabled:true
   ```

   Port `1636` is the standard LDAPS port, but you might not be able to use this port if it is already taken or if you are a regular user. If you must accept SSL-based connections on a port other than `1636`, change the `listen-port` property in the last command to the port number being used.

If you have specified a different value for `-keypass` and `-storepass` when generating the private key in step 1, then you must provide the key password using `dsconfig`:

```
$ dsconfig -D "cn=directory manager" -j pwd-file -X -n \
create-key-manager-provider-key-pin --provider-name JKS --set
 key-pin:<password> --type generic --pin-name
 server-cert
```

For the name of the key pin, provide the same name of the alias of the certificate. This is needed to identify which key pin/password is associated with each certificate in the key manager provider.

Then, update the SSL certificate nickname in the connection handler using the `dsconfig` command as follows:

```
dsconfig -D "cn=directory manager" -j pwd-file -X -n
set-connection-handler-prop --handler-name "LDAPS Connection Handler"
--set ssl-cert-nickname:server-cert
```

For detailed information about keystores and truststores, see Configuring Key Manager Providers and Configuring Trust Manager Providers, respectively.

6. The server should now have a second listener that accepts SSL-based client connections. Test the configuration with the `ldapsearch` command, for example:

```
$ ldapsearch --port 1636 --useSSL --baseDN "" --searchScope base "(objectClass=*)"
```

You are prompted to trust the server's certificate. On typing `yes`, the root DSE entry should be returned.

26.2 Configuring Key Manager Providers

Key manager providers provide access to the certificate that should be used by the directory server when performing SSL or StartTLS negotiation.

Configuring key manager providers are covered in this section:

- Overview of Key Manager Provider
- Using JKS Key Manager Provider
- Using the PKCS #12 Key Manager Provider
- Overview of PKCS #11 Key Manager Provider
- Overview of Hardware-Based Key Manager Provider
- About Replacing a Certificate in a Production Server
- Configuring Key Managers Using OUDSM

For more information, see "Key Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.2.1 Overview of Key Manager Provider

Oracle Unified Directory supports keystore formats for certain key manager providers.

Key manager providers are listed below:

- JKS keystore, which is the default keystore format used by Java Secure Socket Extension (JSSE)
- PKCS #12 file
- Hardware-based devices such as a hardware security module (HSM) or cryptographic accelerator
- PKCS #11 device, which is a specific hardware-based key manager provider

> **Note:**
>
> PKCS #11 is not supported for use with a proxy server instance.

The following sections describe the process for configuring Oracle Unified Directory to use these key manager providers.

The administration connector is an LDAPS connector. As with all SSL-based connectors, the administration connector requires a key manager. Oracle Unified Directory provides a dedicated key manager for the administration connector, that is enabled by default. For more information, see Managing Administration Traffic to the Server.

26.2.2 Using JKS Key Manager Provider

The JKS keystore is the default keystore used by most JSSE implementations, and is the preferred keystore type in many environments. To configure the server to use this keystore type, you must first obtain a JKS keystore that contains a valid certificate. To do this, you can either generate a self-signed certificate or issue a certificate signing request to an existing Certificate Authority (CA) and import the signed certificate.

All of the steps described here require the use of the `keytool` utility, which is provided with the Java runtime environment. This utility is typically found in the `bin` directory below the root of the Java installation. For more information about using the `keytool` utility, see the official Java documentation (`http://download.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html`).

Using the JKS key manager provider involves the following:

- Generating the Private Key
- Self-Signing the Certificate
- Signing the Certificate Using an External Certificate Authority
- Configuring the JKS Key Manager Provider

26.2.2.1 Generating the Private Key

Whether you use a self-signed certificate or generate a certificate signing request, you must first generate a private key. You can do this using the `keytool` utility with the `-genkeypair` option. The following arguments can be used with this option:

Table 26-1 Private Key arguments

Arguments	Description
`-alias` *alias*.	Specifies the name that should be used to refer to the certificate in the keystore. The default name used by server is `server-cert`.
`-keyalg` *algorithm*	Specifies the algorithm that should be used to generate the private key. This should almost always be `rsa`.
`-dname` *subject*.	Specifies the subject to use for the certificate. The subject typically contains at least a `CN` attribute, which is the fully-qualified name of the system on which the certificate will be installed, an `O` attribute that specifies the name of the organization (or company), and a `C` attribute that specifies the country in which the certificate will be used.
`-keystore` *path*.	Specifies the path to the file that contains the private key information. The default keystore path used by the directory server is `config/keystore.jks`. This may be an absolute path or a path that is relative to the Oracle Unified Directory instance root. Changes to this property will take effect the next time that the key manager is accessed.
`-keypass` *password*.	Specifies the password that should be used to protect the private key in the keystore. If the password is not provided, you will be prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storepass` *password*.	Specifies the password that should be used to protect the contents of the keystore. If the password is not provided, you will be prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storetype` *type*.	Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

Use the `keytool -genkeypair` command to create a private key. You will be prompted to enter the required passwords.

```
$ keytool -genkeypair -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore config/keystore.jks -storetype JKS
```

26.2.2.2 Self-Signing the Certificate

If the certificate is to be self-signed, use the `-selfcert` option. The most important arguments for use with this option include:

Table 26-2 Self-signed Certificate options

Arguments	Description
`-alias` *alias*.	Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkeypair` option.
`-validity` *days*.	Specifies the length of time in days that the certificate should be valid. The default validity is 90 days.
`-keystore` *path*.	Specifies the path to the keystore file. The file will be created if it does not already exist.

Table 26-2 (Cont.) Self-signed Certificate options

Arguments	Description
`-keypass` *password*	Specifies the password that should be used to protect the private key in the keystore. If this is not provided, then you will be interactively prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storepass` *password*.	Specifies the password that should be used to protect the contents of the keystore. If this is not provided, then you will be interactively prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storetype` *type*.	Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

Use the `keytool -selfcert` command to generate a self-signed certificate. You will be prompted to provide the required passwords.

```
$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore config/keystore.jks -storetype JKS
```

26.2.2.3 Signing the Certificate Using an External Certificate Authority

If the certificate is to be signed by an external certificate authority, you must first generate a certificate signing request (CSR) using the `-certreq` option. The CSR can be submitted to a certificate authority to be signed. The method for doing this, and the method for obtaining the signed certificate, might vary from one certificate authority to another.

1. Use the `-certreq` option to generate a certificate signing request. You will be prompted to provide the required passwords.

   ```
   $ keytool -certreq -alias server-cert -file /tmp/server-cert.csr \
     -keystore config/keystore.jks -storetype JKS
   ```

   The arguments used with this command are as follows:

   Table 26-3 -certreq option arguments

Arguments	Description
`-alias` *alias*.	Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkeypair` option.
`-file` *path*.	Specifies the path to the file to which the CSR should be written. If this is not provided, the request will be written to standard output.
`-keystore` *path*.	Specifies the path to the keystore file. The file will be created if it does not already exist.

Table 26-3 (Cont.) -certreq option arguments

Arguments	Description
`-keypass` *password*.	Specifies the password that should be used to protect the private key in the keystore. If this is not provided, you will be interactively prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storepass` *password*.	Specifies the password that should be used to protect the contents of the keystore. If this is not provided, you will be interactively prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storetype` *type*.	Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`. This is an optional parameter.

2. Send the certificate request to an external certificate authority. The certificate authority will send you a signed certificate file. Save the file in `/tmp/server-cert.txt`.
3. After receiving the signed certificate from the Certificate Authority, use the `-importcert` option to import it into the keystore.

> **Note:**
>
> If the certificate authority provides you an Intermediate Certificate, then you must also import the Intermediate Certificate into the keystore.

```
$ keytool -importcert -alias server-cert -file /tmp/server-cert.cert \
  -keystore config/keystore.jks -storetype JKS
```

The arguments used with this command are as follows:

Table 26-4 importcert command arguments

Arguments	Description
`-alias` *alias*.	Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkeypair` option.
`-file` *path*	Specifies the path to the file containing the signed certificate. The file should be in either the DER-encoded binary format or the base64-encoded ASCII format as described in RFC 1421 (`http://www.ietf.org/rfc/rfc1421.txt`.
`-keystore` *path*.	Specifies the path to the keystore file. The file will be created if it does not already exist.

Table 26-4 (Cont.) importcert command arguments

Arguments	Description
`-storepass` *password*	Specifies the password that should be used to protect the contents of the keystore. If this is not provided, then you will be interactively prompted for it. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.
`-storetype` *type*.	Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`. This is an optional parameter.

26.2.2.4 Configuring the JKS Key Manager Provider

When you have created a JKS keystore containing a signed certificate (whether self-signed or signed by an external CA), you can configure the server to use that keystore by configuring a key manager provider entry for that keystore.

This example uses `dsconfig` to configure the properties of the default JKS key manager provider. For details about all the properties of the key manager provider, see "File Based Key Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

Use the `dsconfig` command to configure the key manager provider entry.

```
dsconfig -D "cn=Directory Manager" -j pwd-file -X -n \
  set-key-manager-provider-prop --provider-name "JKS" \
  --set enabled:true --set "key-store-type:JKS" \
  --set "key-store-file:config/keystore.jks" \
  --set "key-store-pin:<key-store-pwd>" \
```

> **Note:**
>
> You need to use the argument, **--reset key-store-pin-file** along with other arguments.

If you have specified a different value for `-keypass` and `-storepass` when generating the private key in step 1 of Generating the Private Key, you must provide the key password using dsconfig. For example:

```
dsconfig -D "cn=directory manager" -j pwd-file -X -n \
create-key-manager-provider-key-pin --provider-name JKS
--set key-pin:<key password> --type generic --pin-name server-cert
```

> **Note:**
>
> You need to use the argument, **--reset key-store-pin-file** along with other arguments.

Important: When you provide the name of the key pin, use the same name as the alias of the certificate. The key pin name and the certificate alias name must be identical to identify which key pin/password is associated with each certificate in the key manager provider.

26.2.3 Using the PKCS #12 Key Manager Provider

PKCS #12 is a standard format for storing certificate information, including private keys. Oracle Unified Directory can use a PKCS #12 file as a certificate keystore if it includes the private key for the certificate.

Because PKCS #12 is a common format for storing certificate information, you might already have a certificate in this format, or the certificate authority (CA) that you use might create certificates in this form. In some cases, it might also be possible to convert an existing certificate into PKCS #12 format. For example, if you already have a certificate in a Network Security Services (NSS) certificate database, then the NSS `pk12util` tool can import it.

The following example uses the `pk12util` tool to export a certificate named `server-cert` contained in the database `../../alias/slapd-config-key3.db` to a PKCS #12 file, `/tmp/server-cert.p12`:

```
$ ./pk12util -n server-cert -o /tmp/server-cert.p12 \
  -d ../../alias -P "slapd-config-"
```

To create a new certificate in PKCS #12 format, use the procedure described in Using JKS Key Manager Provider for obtaining a certificate in a JKS keystore. The only difference in the process is that you should use `-storetype PKCS12` instead of `-storetype JKS` when you invoke the `keytool` commands. For example, to create a self-signed certificate in a PKCS #12 file, use the following commands:

```
$ keytool -genkeypair -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore config/keystore.p12 -storetype PKCS12

$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore config/keystore.p12 -storetype PKCS12
```

As with JKS, the server provides a template key manager provider for use with PKCS #12 certificate files that uses the same set of configuration attributes as the configuration entry for the JKS key manager provider. The only differences are that the value of the `key-store-type` attribute must be `PKCS12`, and the `key-store-file` attribute should refer to the location of the PKCS #12 file rather than a JKS keystore. The following example uses `dsconfig` to configure the PKCS #12 keystore manager provider:

```
$ dsconfig -D "cn=directory manager" -j pwd-file -X -n\
  set-key-manager-provider-prop --provider-name "PKCS12" --set enabled:true \
  --set java-class:org.opends.server.extensions.FileBasedKeyManagerProvider \
  --set enabled:true --set "key-store-type:PKCS12" \
  --set "key-store-file:/config/keystore.p12" \
  --set "key-store-pin:secret" \
```

For a complete list of configurable properties, see "File Based Key Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.2.4 Overview of PKCS #11 Key Manager Provider

PKCS #11 is a standard interface used for interacting with devices capable of holding cryptographic information and performing cryptographic functions.

The PKCS #11 interface has two common uses of interest for the directory server:

- Cryptographic accelerators use this interface to allow products to offload their cryptographic processing to an external board (or in some cases, a special module inside

the system's CPU or a framework inside the OS kernel), which might provide better performance for those operations.

- Hardware security modules (HSMs) use this interface to provide a secure repository for storing key information. This significantly reduces the likelihood that sensitive key information will be exposed and helps protect the overall integrity of the secure communication mechanisms.

> **Note:**
>
> The PKCS #11 format is not supported for use with a proxy server instance.

Oracle Unified Directory provides PKCS #11 support that, currently, has been tested and verified only on systems running at least Solaris 10 (on SPARC and x86/x64 systems) with the Solaris OS cryptographic framework. Any devices that plug into this Solaris cryptographic framework should be supported in this manner — including the *softtoken* device, which is simulated in software and is therefore available on all systems supporting the Solaris cryptographic framework, regardless of whether they have a hardware device providing PKCS #11 support.

If you do have a third-party PKCS #11 device installed in a Solaris system, it is likely that the Solaris OS cryptographic framework is already configured to access that device. However, if you simply use the software token or if you run on a Sun Fire T1000 or T2000 system and want to take advantage of the cryptographic processor included in the UltraSPARC—T1 CPU, you will likely need to initialize the PKCS #11 interface. This should first be accomplished by choosing a PIN to use for the certificate store, which can be done with this command:

```
$ pktool setpin
```

This command prompts you for the current passphrase. If you have not yet used the Solaris OS cryptographic framework, the default passphrase is `changeme`. You are then prompted twice for the new password.

> **Note:**
>
> This step should be done while you are logged in as the user or as the role that will be used to run the directory server, because each user might have a different set of certificates.

At this point, it should be possible to use the Java `keytool` utility to interact with the Solaris cryptographic framework through PKCS #11. This will work much in the same way as it does when working with JKS or PKCS#12 keystores, with the following exceptions:

- The value of the `-keystore` argument must be `NONE`.
- The value of the `-storetype` argument must be `PKCS11`.
- You should not use the `-keypass` argument, and the tool will not prompt you for that password interactively if you do not provide it.
- The value of the `-storepass` argument must be the passphrase that you chose when using the `pktool setpin` command. Alternately, if you do not provide this argument on the command line, this is the password that you should enter when prompted.

For example, the following commands use the PKCS #11 interface to generate a self-signed certificate through the Solaris cryptographic framework:

```
$ keytool -genkeypair -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore NONE -storetype PKCS11

$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore NONE -storetype PKCS11
```

When the certificate is installed in the PKCS #11 keystore, the directory server must be configured to use that keystore. Configure the PKCS #11 keystore provider in the same way as the entry for the JKS and PKCS#12 keystore manager providers, except that the `key-store-file` attribute is not included. However, a PIN is still required and is provided either directly, in a PIN file, through a Java property, or through an environment variable.

The following example uses `dsconfig` to configure the PKCS #11 key manager provider:

```
$ dsconfig -D "cn=directory manager" -j pwd-file -X -n \
  set-key-manager-provider-prop --provider-name "PKCS11" --set enabled:true \
  --set enabled:true --set "key-store-type:PKCS11" \
  --set "key-store-file:/config/keystore" \
  --set "key-store-pin:secret" \
```

> **Note:**
>
> You need to use the argument, **--reset key-store-pin-file** along with other arguments.

For a complete list of configurable properties, see "PKCS11 Key Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.2.5 Overview of Hardware-Based Key Manager Provider

You can create a key manager provider of type Hardware-Based. The Hardware Based Key Manager Provider enables the server to access the private key information through a generic hardware-based key store. This standard interface is used by cryptographic accelerators and hardware security modules.

Cryptographic accelerators use this interface to allow products to offload their cryptographic processing to an external board (or in some cases, a special module inside the system's CPU or a framework inside the OS kernel), which might provide better performance for those operations.

Hardware security modules (HSMs) use this interface to provide a secure repository for storing key information. This significantly reduces the likelihood that sensitive key information will be exposed and helps protect the overall integrity of the secure communication mechanisms.

Before Oracle Unified Directory can use a hardware-based key manager provider, the Java Virtual Machine used by Oracle Unified Directory must be configured to integrate with the HSM. To verify that the HSM is properly configured, use keytool to read its contents.

To specify the PIN to be used to access the key store, configure one of the following: a Java property, an environment variable, the PIN value itself, or the path to a file containing the PIN in clear text.

The following example uses `dsconfig` to configure the Hardware-Based key manager provider:

```
$ dsconfig -D "cn=directory manager" -j pwd-file -X -n \
  set-key-manager-provider-prop --provider-name "Hardware-Based" \
  --set enabled:true --set "key-store-type:Hardware-Based" \
  --set "key-store-file:/config/keystore" \
  --set "key-store-pin:secret" \
  --reset key-store-pin-file
```

For a complete list of configurable properties, see "Hardware-Based Key Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.2.6 About Replacing a Certificate in a Production Server

In a production server, to replace a certificate, you need to request the new certificate. The `key-manager-provider` property of the SSL-based connection handler (named "LDAPS" by default) specifies the keystore manager that must be used for security.

To replace a certificate in a production server, request the new certificate and configure the appropriate key manager provider, as described in Using JKS Key Manager Provider, Using the PKCS #12 Key Manager Provider, or Overview of PKCS #11 Key Manager Provider.

The default value of the `key-manager-provider` property is "JKS", which means that the SSL connection handler uses the JKS key manager provider by default. If you are using a different key manager provider, change this property of the SSL connection handler accordingly.

The server needs to be restarted after the new certificate is installed.

26.2.7 Configuring Key Managers Using OUDSM

You can configure key manager configuration by using OUDSM.

Perform the following steps to manage the key manager configuration:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Under General Configuration, expand the **Key Managers** item.
4. Select the key manager you want to configure.

   The configurable properties of the key manager are displayed in the right hand pane.
5. Edit the key manager configuration, as required, and click **Apply** to save your changes.

26.3 Configuring Trust Manager Providers

Oracle Unified Directory uses trust manager providers to determine whether to trust a certificate that is presented to it. Trust managers serve an important role in the overall security of the system by ensuring that the peer (the system at the other end of the connection, whether it is an inbound connection from a client or an outbound connection to another server) is who it claims to be.

The following topics covers information on trust manager providers and using them:

- Overview of Certificate Trust Mechanisms
- About Blind Trust Manager Provider
- Using the JKS Trust Manager Provider

- Using the PKCS #12 Trust Manager Provider
- Configuring Trust Managers Using OUDSM

26.3.1 Overview of Certificate Trust Mechanisms

A trust manager provider can improve security whenever SSL or StartTLS is used by thwarting attempts to use forged certificates and foiling man-in-the-middle attacks.

The two primary use cases for trust manager providers are as follows:

- Inbound connections: a client presents its own certificate to the server during the SSL or StartTLS negotiation process, potentially for use in SASL EXTERNAL authentication.
- Outbound connections: the server attempts to establish an SSL-based connection to an external system, for example for the purpose of synchronization or for proxied or chained operations.

The trust manager has no impact on the strength of the encryption, so only the server and its peer will be able to understand the communication. Any third-party observer will be unable to decipher the exchange. The trust manager is responsible for ensuring that the peer is who it claims to be so that confidential information is not inadvertently exposed to one peer masquerading as another.

The trust manager considers several factors to determine whether a peer certificate should be trusted. This topic describes some common criteria that are taken into account during this process.

One of the simplest trust mechanisms is the validity period for the certificate. All certificates have a specific window during which they should be considered valid, bounded by "notBefore" and "notAfter" time stamps. If the current time is beyond the "notAfter" time stamp, the certificate is expired and trust managers reject it. Similarly, certificates are also typically rejected if the current time is before the "notBefore" time stamp. Most often, the "notBefore" time stamp is set to the time that the certificate was signed, but there are cases in which a certificate might be issued that is not immediately valid. In those cases, it is important to ensure that the peer is not granted access too early.

Another very important factor in deciding whether to trust a peer certificate is the peer certificate chain. When one system presents its certificate to another, it does not present its certificate only, but a chain of certificates that describes all entities involved in the process. When a trust manager is attempting to determine whether to trust a peer, the trust manager first looks in its trust store to determine whether it contains the peer certificate. If that certificate is found, the peer will be trusted (barring rejection for another reason, such as being outside the validity period). If the peer's certificate is not found, the trust manager looks at the next certificate in the chain, which will be the certificate that was used to sign the peer's certificate (also called the issuer certificate). If the trust store contains the issuer's certificate, the server will trust that issuer certificate and will also implicitly trust any certificate that it has signed. This process continues up the certificate chain (looking at the certificate that signed the issuer certificate, and so on) until one of the certificates is found in the trust store or until the root of the chain is reached (in which case, the root certificate will be self-signed and therefore will be its own issuer). If none of the certificates in the peer chain is contained in the trust store, the peer's certificate is rejected.

This process makes it much easier to manage an environment with a large number of certificates (for example, one in which there is a large number of servers or in which many clients use SASL EXTERNAL authentication). It is not necessary for the trust store to have each individual peer certificate. The trust store can contain only one of the certificates in the peer chain. For example, if all of the certificates that might legitimately be presented to the

server were signed by the same issuer, then it is necessary to have only that issuer's certificate in the trust store to implicitly trust any of the peers.

In some environments, there might be other elements taken into account when deciding to trust a peer certificate chain. For example, there might be a certificate revocation list (CRL) that contains a list of all of the certificates that have been revoked and should no longer be considered valid even if they are still within their validity period and were signed by a trusted issuer. This can be useful, for example, if the certificate belonged to an employee that has left the company or if the private key for the certificate has been compromised. The Online Certificate Status Protocol (OCSP, as described in RFC 2560 (`http://www.ietf.org/rfc/rfc2560.txt`) also provides a similar mechanism, in which the trust manager might ask an OCSP server whether a given certificate is still valid. Oracle Unified Directory currently does not support using CRLs or OCSP when attempting to determine whether a peer certificate chain should be trusted.

The administration connector is an LDAPS connector. As with all SSL-based connectors, the administration connector requires a trust manager. Oracle Unified Directory provides a dedicated trust manager for the administration connector, that is enabled by default. For more information, see Managing Administration Traffic to the Server.

26.3.2 About Blind Trust Manager Provider

The blind trust manager provider is a simple provider that trusts any certificate that is presented to it. It does not look at the expiration date, who signed the certificate, the subject or alternate names, or any other criteria.

Oracle Unified Directory provides a blind trust manager provider that is disabled by default. You can enable the provider by changing the value of the `enabled` attribute to `true`. The blind trust manager provider does not require any other configuration attributes.

> **Note:**
>
> The blind trust manager provider is not supported with a proxy server instance.

The following example uses `dsconfig` to configure the blind trust manager provider:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-trust-manager-provider-prop --provider-name "Blind Trust"
```

For a list of the configurable properties, see the "Blind Trust Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

> **Caution:**
>
> The blind trust manager provider is provided as a convenience for testing purposes only and should never be used in a production server, especially one that is configured to allow SASL EXTERNAL authentication. If a client attempts to use SASL EXTERNAL to authenticate to using a certificate and the server blindly accepts any certificate that the client presents, the user can create a self-signed certificate that allows it to impersonate any user in the directory.

26.3.3 Using the JKS Trust Manager Provider

Just as the JKS keystore can be used to provide the key material for a key manager provider, it can also be used to provide information that can used by trust manager providers. In general, using a JKS file as a trust store is similar to using it as a keystore. However, because private key information is not accessed when the file is used as a trust store, there is generally no need for a PIN when accessing its contents.

When the JKS trust manager provider determines whether to trust a given peer certificate chain, it considers two factors:

- Is the peer certificate within the validity period?
- Is any certificate in the chain contained in the trust store?

If the peer certificate is not within the validity period or none of the certificates in the peer certificate chain are contained in the trust store, the JKS trust manager rejects that peer certificate.

Use the `keytool -importcert` utility to import certificates into a JKS trust store. The `-importcert` option uses these arguments:

Table 26-5 -importcert options

Arguments	Description
`-alias alias`.	Specifies the name to give to the certificate in the trust store. Give each certificate a unique name, although the nickname is primarily for managing the certificates in the trust store and has no impact on whether a certificate is trusted.
`-file path`.	Specifies the path to the file containing the certificate to import. The file can be in either DER format or in base64-encoded ASCII format, as described in RFC 1421 (`http://www.ietf.org/rfc/rfc1421.txt`).
`-keystore path`.	Specifies the path to the file containing the trust information. The default `-keystore` path used by the directory server when specified for the truststore is `config/truststore.jks`. This may be an absolute path or a path that is relative to the Oracle Unified Directory instance root. Changes to this property will take effect the next time that the trust manager is accessed.
`-storetype type`.	Specifies the format of the trust store file. For the JKS trust manager, this must be `JKS`
`-storepass password`.	Specifies the password used to protect the contents of the trust store. If the trust store file does not exist, this value is the password to assign to the trust store, and must be used for future interaction with the trust store. If this option is not provided, the password is interactively requested from the user. This is an optional parameter and is not recommended to be used considering the storage of clear text password in the command-line history.

The following command provides an example of importing the Root CA certificate into a JKS trust store. If the trust store does not exist, this command creates the trust store before importing the certificate.

```
$ keytool -importcert -alias rootca -file /tmp/rootca_cert.txt
  -keystore config/truststore.jks -storetype JKS
```

Oracle Unified Directory provides a template JKS trust manager provider. Use `dsconfig` to configure the following properties of the JKS trust manager provider:

Table 26-6 JKS trust manager provider properties

Properties	Description
`enabled`.	Indicates whether the JKS trust manager provider is enabled. The JKS trust manager provider is not available for use by other server components unless the value of this property is `true`.
`trust-store-file`.	The path to the trust store file, which is typically `config/truststore.jks`, although an alternate file can be used if needed. The value of this property can be either an absolute path or a path that is relative to the *INSTANCE_DIR*
`trust-store-type`.	The format of the trust store. For the JKS trust store provider, the value of this property is `JKS`

The following example uses `dsconfig` in interactive mode to configure the JKS trust manager provider:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-trust-manager-provider-prop --provider-name "JKS" \
--set enabled:true --set "trust-store-type:JKS" \
--set trust-store-file:config/truststore.jks
```

For a list of the configurable properties, see the "File Based Trust Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.3.4 Using the PKCS #12 Trust Manager Provider

The PKCS #12 trust manager provider is primarily useful if you already have the peer or issuer certificates to be used in a PKCS #12 file. If you do not have the certificates in this format, use the JKS trust manager provider instead. The Java `keytool` utility does not currently support importing trusted certificates (that is, those with just a public key and no private key information) into a PKCS #12 file.

Oracle Unified Directory provides a template PKCS #12 trust manager provider. Use `dsconfig` to configure the following properties of the PKCS #12 trust manager provider:

Table 26-7 Properties of PKCS #12 trust manager provider

Property	Description
`enabled`	Indicates whether the PKCS #12 trust manager provider is enabled. The trust manager provider is not available for use by other server components unless this property has a value of true
`trust-store-type`.	Specifies the format of the trust store. For the PKCS #12 trust manager provider, the value is `PKCS12`.
`trust-store-file`.	Specifies the path to the trust store file, which is typically `config/truststore.p12`, although an alternate file can be used if needed. The value of this property can be either an absolute path or a path that is relative to the *INSTANCE_DIR*.

A PIN might be required to access the contents of the PKCS #12 file. In this case, you must use the `trust-store-pin` configuration attribute to provide the password. (Currently, the password must be provided in clear text.)

> **Note:**
>
> From April 2021 bundle patch release onward, the `trust-store-pin-file`, `trust-store-pin-property`, and `trust-store-pin-environment-variable` configuration attributes are no longer supported. The PIN value determined from these three configuration attributes is moved to the `trust-store-pin` attribute after you upgrade to the April 2021 bundle patch. The `trust-store-pin` contains the PIN needed to access the trust store directly.

The following example uses `dsconfig` in interactive mode, to configure the PKCS #12 trust manager provider:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-trust-manager-provider-prop --provider-name "PKCS12"
```

For a list of the configurable properties, see the "File Based Trust Manager Provider Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.3.5 Configuring Trust Managers Using OUDSM

To configure the Trust Managers by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Under General Configuration, expand the **Trust Managers** item.
4. Select the trust manager you want to configure.

   The configurable properties of the trust manager are displayed in the right hand pane.
5. Edit the trust manager configuration, as required, and click **Apply** to save your changes.

26.4 Configuring Certificate Mappers

A *certificate mapper* examines a certificate presented by a client and maps it to the user in the directory that should be associated with that certificate. Certificate mappers are configured for directory server instances only - not for proxy or gateway instances.

Certificate mappers are primarily used in the context of processing SASL EXTERNAL authentication, where the client wants to authenticate to the server using its SSL certificate rather than a password or some other form of credentials.

Oracle Unified Directory provides the following certificate mappers by default:

- Using Subject Equals DN Certificate Mapper
- Using Subject Attribute to User Attribute Certificate Mapper
- Using Subject DN to User Attribute Certificate Mapper
- Using Subject Alternative Name To User Attribute Certificate Mapper
- Using Fingerprint Certificate Mapper

You can also create a custom certificate mapper to suit the requirements of your deployment.

A certificate mapper is defined either at the global server configuration level, or at the network group level. If a certificate mapper is defined for the network group, that certificate mapper overrides what is defined in the global server configuration. If no certificate mapper is defined for a network group, the global certificate mapper is used. To define the certificate mapper that should be used, set the `certificate-mapper` property of the global configuration, or the network group.

The examples in this section use the `dsconfig` command to modify certificate mappers. The `dsconfig` command accesses the server configuration over SSL, using the administration connector. For more information, see Managing the Server Configuration Using `dsconfig`.

26.4.1 Using Subject Equals DN Certificate Mapper

The Subject Equals DN certificate mapper is a simple certificate mapper that expects the subject of the client certificate to be exactly the same as the distinguished name (DN) of the corresponding user entry. Using this certificate mapper is easy because there are no configuration attributes associated with it. However, this mapper is not suitable for many environments because certificate subjects and user DNs are often different.

The server uses the Subject Equals DN certificate mapper by default. To change the certificate mapper that is used by the server, set the appropriate global configuration property by using `dsconfig`. The following command changes the certificate mapper that the server uses from Subject Equals DN to Subject Attribute to User.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-global-configuration-prop \  --set certificate-mapper:"Subject Attribute to User
Attribute"
```

You cannot disable the Subject Equals DN certificate mapper if it is referenced by the global server configuration. To disable the mapper, you must change the default certificate mapper, as described previously.

26.4.2 Using Subject Attribute to User Attribute Certificate Mapper

The Subject Attribute to User Attribute certificate mapper attempts to map a client certificate to a user entry based on a set of attributes that they have in common. In particular, it takes the values of a specified set of attributes from the certificate subject and attempts to locate user entries that contain those same values in a corresponding set of attributes.

Use `dsconfig` to set the properties of this certificate mapper:

- `subject-attribute-mapping`. A multi-valued property that maps attributes from the certificate subject to attributes in user entries. Values for this attribute consist of the name of the attribute in the certificate subject followed by a colon and the name of the corresponding attribute in the user's entry. For example, the value `e:mail` maps the `e` attribute from the certificate subject to the `mail` attribute in user entries. At least one attribute mapping must be defined. The default mappings are `e:mail` and `cn:cn`.
- `user-base-dn`. A multi-valued property that specifies the set of base DNs below which the server should look for matching entries. If this attribute has no value, the server searches below all public naming contexts.

The following example uses `dsconfig` to configure the Subject Attribute to User Attribute certificate mapper, specifying that the server should search only below `ou=people,dc=example,dc=com`:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-certificate-mapper-prop \
```

```
--mapper-name "Subject Attribute to User Attribute" \
--set user-base-dn:ou=people,dc=example,dc=com
```

If multiple attribute mappings are defined, the server combines them with an AND search. For example, if two mappings are defined `cn:cn` and `e:mail`, and the server is presented with a certificate that has a subject of `E=john.doe@example.com,CN=John Doe,O=Example Corp,C=US`, then it generates a search filter of `(&(cn=John Doe)(mail=john.doe@example.com))`. Any attribute for which a mapping is defined but is not contained in the certificate subject is not included in the generated search filter. All attributes that can be used in generated search filters should have corresponding indexes in all remote LDAP databases that can be searched by this certificate mapper.

For the mapping to be successful, the generated search filter must match exactly one user in the directory (within the scope of the base DNs for the mapper). If no users match the generated criteria or if multiple users match, the mapping fails.

26.4.3 Using Subject DN to User Attribute Certificate Mapper

The Subject DN to User Attribute certificate mapper attempts to establish a mapping by searching for the subject of the provided certificate in a specified attribute in user entries. In this case, you must ensure that user entries are populated with the subjects of the certificates associated with those users. However, this process might possibly be automated in the future with a plug-in that automatically identifies any certificates contained in a user entry and adds the subjects of those certificates to a separate attribute.

Use `dsconfig` to set the properties of this certificate mapper:

- `subject-attribute`. This is a single-valued attribute whose value is the name of the attribute type that should contain the certificate subject in user entries. This attribute must be defined in the server schema, and it should be indexed for equality in all back ends that might be searched.

  The subject DN of the certificate received by the server will not contain any spaces between its RDN components, even though the certificate might have been created with them. The value of the `subject-attribute` in the user entries must also not contain any spaces between the RDN components, so that they will correctly match the subject DN of the received certificate. For example, if the original certificate looks like:

  ```
  keytool -printcert -file cert.002
  Owner: CN=test,  O=Test Certificate
  Issuer: CN=test,  O=Test Certificate
  Serial number: 49b55976
  Valid from: Mon Mar 09 19:01:26 MET 2009 until: Sat Mar 08 19:01:26 MET 2014
  Certificate fingerprints:
           MD5:  5E:08:78:36:DF:25:F4:6C:43:9E:7B:CF:1F:1E:B9:6B
           SHA1: B7:B9:1C:A0:B0:52:C3:87:3C:C2:70:27:11:6F:5E:58:C5:33:9D:6B
           Signature algorithm name: SHA1withRSA
           Version: 3
  ```

  The subject DN defined in the subject-attribute of the user entry should be:

  ```
  CN=test,O=Test Certificate
  ```

  Note the removal of the space between the RDN components of the `subject-attribute`.

- `user-base-dn`. This is a multivalued attribute that is used to specify the set of base DNs below which the server should look for matching entries. If this is not present, then the server will search below all public naming contexts.

The following example uses `dsconfig` to configure the Subject DN to User Attribute certificate mapper, specifying that the server should search only below `ou=people,dc=example,dc=com`:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-certificate-mapper-prop \
  --mapper-name "Subject DN to User Attribute" \
  --set user-base-dn:ou=people,dc=example,dc=com
```

Although there is no standard attribute for holding the subjects of the certificates that a user might hold, does define a custom attribute type, `ds-certificate-subject-dn`, that can be used for this purpose. This attribute can be added to user entries along with the `ds-certificate-user` auxiliary object class. This attribute is multivalued. If a user has multiple certificates, the attribute should contain the subjects for each of them as separate values.

This attribute is not indexed by default, so if it is to be used, update the corresponding back ends so that they contain an equality index for this attribute.

For the mapping to be successful, the certificate mapper must match exactly one user (within the scope of the base DNs for the mapper). If no entries match or if multiple entries match, the mapping fails.

26.4.4 Using Subject Alternative Name To User Attribute Certificate Mapper

Subject Alternative Name to User Attribute certificate mapper attempts to establish a mapping between Oracle Unified Directory and certificate by fetching the Principal Name (or other names) that is present under subject alternative name extension of the provided certificate. You must ensure that user entries are populated with the Principal Name(or other names) that are present under subject alternative name extension of the certificates, are associated with those users.

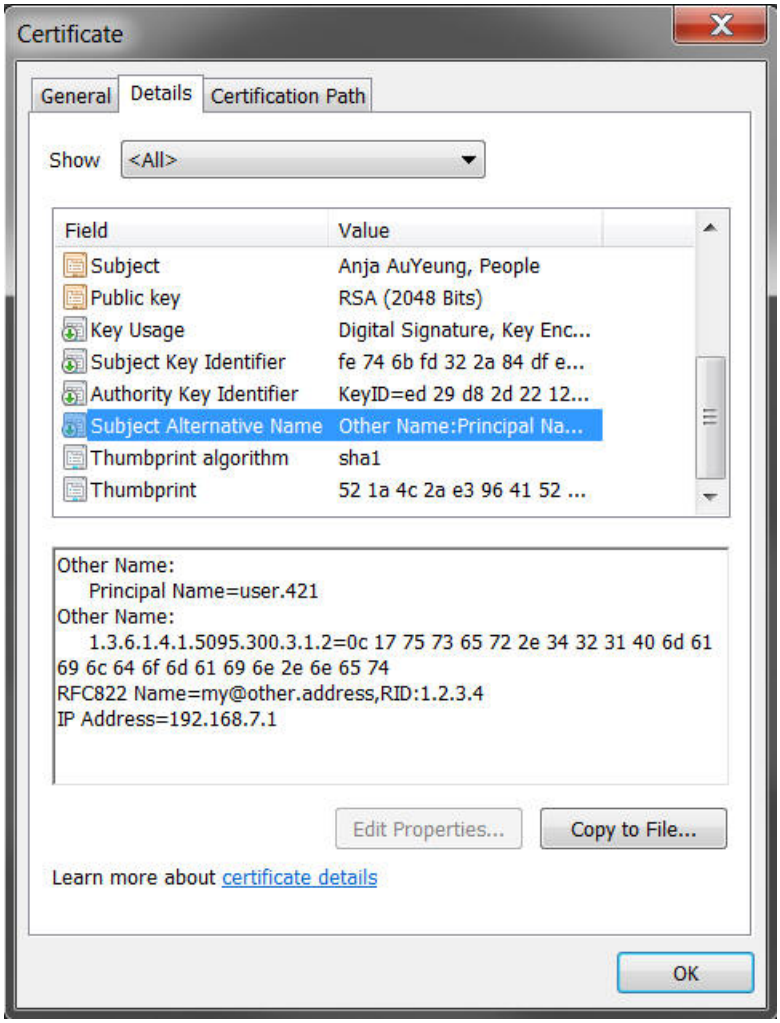
Use `dsconfig` to set the properties of this certificate mapper:

- `subject-alternative-name-attribute-mapping`

  A multi-valued property that maps attributes from the certificate subject to alternative attributes in user entries. Values for this attribute consist of the Principal Name attribute in the certificate subject followed by a colon and the name of the corresponding attribute in the user's entry. For example, the value `user.421` maps to Principal Name and query to OUD is based on the mapping configuration defined in the SAN mapper.

  For example, Figure 26-1 is a certificate containing the Subject Alternative Name To User attribute certificate mapper for smart card login use case having `subject-alternative-name-attribute-mapping` value as `1.3.6.1.4.1.311.20.2.3:cn` where `1.3.6.1.4.1.311.20.2.3= Principal Name` and `cn =user.421`:

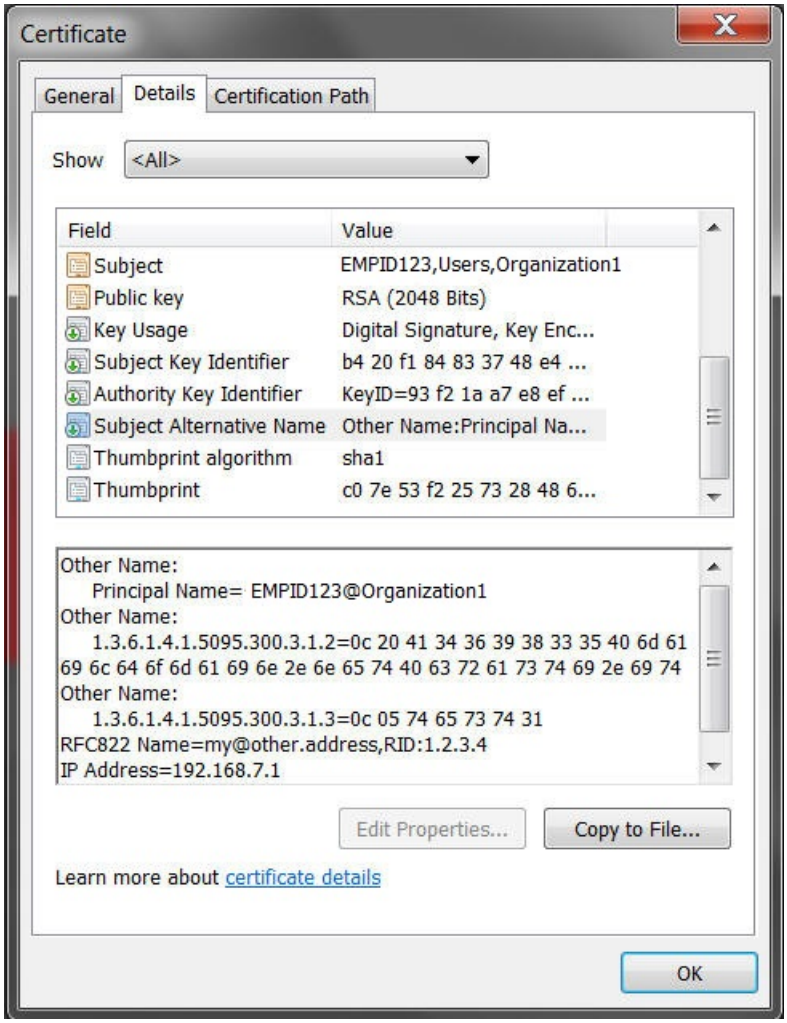
Figure 26-1 Example1: Subject Alternative Name to User Attribute Certificate Mapper



In the below example, the attribute mapping is defined in the mapper is as follows:

In the certificate extension, `subject-alternative-name-attribute-mapping` value is `1.3.6.1.4.1.311.20.2.3:cn@:ou`, where `cn=EMPID123` and `ou=Orgainzation1`

Figure 26-2 Example 2: Subject Alternative Name to User Attribute Certificate Mapper



- `user-base-dn`

  A multi-valued property that specifies the set of base DNs used in this certificate mapper.

Smart card users present their certificate to ESSO during login or authentication which will be passed to Oracle Unified Directory. Oracle Unified Directory has to read this certificate and get the subject or user details which is present under one of the certificate extension called "Subject Alternative Names". The "Subject Alternative Names" has the Principal Name attribute whose oid is 1.3.6.1.4.1.311.20.2.3 and has the actual subject details. This value is read by Oracle Unified Directory and it performs authentication.

> **Note:**
>
> If multiple attribute mappings are defined, the server combines them with an AND search. For example, if two mappings are defined cn:cn and e:mail, and the server is presented with a certificate that has a subject of E=john.doe@example.com,CN=John Doe, O=Example Corp,C=US, then it generates a search filter of (&(cn=John Doe)(mail=john.doe@example.com)). Any attribute for which a mapping is defined, but is not contained in the certificate subject is not included in the generated search filter. All attributes that can be used in generated search filters should have corresponding indexes in all remote LDAP databases that can be searched by this certificate mapper.

You can enable the Subject Alternative Name To User attribute certificate mapper by executing these commands:

- By using `set-certificate-mapper-prop`:

```
$ dsconfig set-certificate-mapper-prop \
 --mapper-name "Subject Alternative Name To User Attribute" \
--set enabled:true -n -X -h localhost -p 1444 -D "cn=directory manager" -j pwdfile
```

- By using `set-global-configuration-prop`:

```
$ dsconfig set-global-configuration-prop \
 --set certificate-mapper:"Subject Alternative Name To User Attribute" -n -X -h
localhost -p 1444 -D "cn=directory manager" -j pwdfile
```

The container structure for smart card login use-case with `subject-alternative-name-attribute-mapping` is explained with the following example:

- `user-base-dn` value `dc=example`, `dc=com` , and `subject-alternative-name-attribute-mapping` value is `1.3.6.1.4.1.311.20.2.3:cn@:ou` where `ou=organization1` and `cn=EMPID123`:

```
com,example, organization1, users,
     EMPID123
     EMPID456
     ......
```

- `user-base-dn` value `ou=users`, `dc=example`, `dc=com` and `subject-alternative-name-attribute-mapping` value is `1.3.6.1.4.1.311.20.2.3:cn@:ou` where `ou=organization1` and `cn=EMPID123`:

```
com, example, users, organization1,
     EMPID123
     EMPID456
     ......
```

You can set `user-base-dn`, and configure the Subject Alternative Name To User Attribute certificate mapper by specifying that the server should search only `dc=example,dc=com`:

```
$ dsconfig set-certificate-mapper-prop \
 --mapper-name "Subject Alternative Name To User Attribute" \
--set user-base-dn:dc=example,dc=com
--hostname localhost --port 1444 --portProtocol LDAP --trustStorePath /<oud-
instance>/OUD/config/admin-truststore
--bindDN "cn=Directory Manager"
--bindPasswordFile pwdfile --no-prompt
```

For the mapping to be successful, the generated search filter must match exactly one user in the directory (within the scope of the base DNs for the mapper). If no users match the generated criteria or if multiple users match, the mapping fails.

26.4.5 Using Fingerprint Certificate Mapper

The Fingerprint certificate mapper attempts to establish a mapping by searching for the MD5 or SHA1 fingerprint of the provided certificate in a specified attribute in user entries.

> **Note:**
>
> JDK 8 adds MD5 in the list of disabled algorithms. This JDK release introduces a new restriction on how MD5 signed JAR files are verified. If the signed JAR file uses MD5, signature verification operations will ignore the signature and treat the JAR as if it were unsigned. To revert this restriction, you must remove MD5 from the list of disabled algorithms by updating the security property, `jdk.jar.disabledAlgorithms`, in the *java.security* file. The *java.security* file is located in `JAVA_HOME/jre/lib/security/java.security`.

In this case, you must ensure that user entries are populated with the certificate fingerprints (in standard hexadecimal notation with colons separating the individual bytes, for example, `07:5A:AB:4B:E1:DD:E3:05:83:C0:FE:5F:A3:E8:1E:EB`). In the future, this process could be automated by a plug-in that automatically identifies any certificates contained in user entries and adds the fingerprints of those certificates to the appropriate attribute.

Use `dsconfig` to set the properties of this certificate mapper:

- `fingerprint-attribute`. Specifies a single-valued attribute whose value is the name of the attribute type that should contain the certificate fingerprint in user entries. This attribute must be defined in the server schema, and it should be indexed for equality in all back ends that can be searched.
- `fingerprint-algorithm`. Specifies which digest algorithm to use to calculate certificate fingerprints. The value is either `MD5` or `SHA1`.
- `user-base-dn`. Specifies a multi-valued attribute that is used to specify the set of base DNs below which the server is to look for matching entries. If this property is not present, then the server searches below all public naming contexts.

The following example uses `dsconfig` to configure the Fingerprint certificate mapper, specifying that the server should search only below `ou=people,dc=example,dc=com`:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-certificate-mapper-prop \
  --mapper-name "Fingerprint Mapper" \
  --set user-base-dn:ou=people,dc=example,dc=com
```

Although there is no standard attribute for holding certificate fingerprints, does define a custom attribute type, `ds-certificate-fingerprint`, that can be used for this purpose. This attribute can be added to user entries along with the `ds-certificate-user` auxiliary object class. This attribute is multi-valued, and if a user has multiple certificates, then it should contain the fingerprints for each of them as separate values. However, this attribute type is not indexed by default in any of the server back ends, so if it is to be used, add the corresponding equality index to all appropriate back ends.

For the mapping to be successful, the certificate mapper must match exactly one user (within the scope of the base DNs for the mapper). If no entries match or if multiple entries match, the mapping fails.

26.5 Configuring SSL and StartTLS for LDAP and JMX

When you have configured Oracle Unified Directory with at least one enabled key manager provider and at least one enabled trust manager provider, you can enable SSL and StartTLS for the connection handlers.

The examples in this section use the `dsconfig` command to modify the server configuration. The `dsconfig` command accesses the server configuration over SSL through the administration connector. As such, the relevant connection options must be specified, including how the SSL certificate is trusted. These examples use the `-X` option to trust all certificates.

This section includes the following topics:

- Configuring the LDAP and LDAPS Connection Handlers
- About JMX Connection Handler

26.5.1 Configuring the LDAP and LDAPS Connection Handlers

The LDAP connection handler is responsible for managing all communication with clients using LDAP. By default, the LDAP protocol does not specify any form of security for protecting that communication, but you can configure it to use SSL or to allow the use of the StartTLS extended operation.

The server configures two connection handlers that can be used for this purpose. While the LDAP connection handler entry is enabled by default and is used to perform unencrypted LDAP communication, it can also be configured to support StartTLS. For information, see Enabling StartTLS Support.

The LDAPS connection handler entry is disabled, but the default configuration is set up for enabling SSL-based communication. For more information, see Enabling SSL-Based Communication.

The following topics describe how to configure LDAP and LDAPS connection handler parameters with `dsconfig`:

- Enabling a Connection Handler
- Specifying a Connection Handler's Listening Port
- Specifying a Connection Handler's Authorization Policy
- Specifying a Nickname for a Connection Handler's Certificate
- Specifying a Connection Handler's Key Manager Provider
- Specifying a Connection Handler's Trust Manager Provider
- Enabling StartTLS Support
- Enabling SSL-Based Communication
- Specifying Protocol Version and Cipher Suites in a Connection Handler

26.5.1.1 Enabling a Connection Handler

Set the `enabled` property of the connection handler to `true`.

This example enables the LDAP connection handler.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set enabled:true
```

26.5.1.2 Specifying a Connection Handler's Listening Port

Set the `listen-port` property of the connection handler.

The `listen-port` property specifies the port number to use when communicating with the server through this connection handler.

The standard port to use for unencrypted LDAP communication (or LDAP using StartTLS) is `389`, and the standard port for SSL encrypted LDAP is `636`. However, it might be desirable or necessary to change this in some environments, for example, if the standard port is already in use or if you run on a UNIX system as a user without sufficient privileges to bind to a port below `1024`.

In UNIX-like systems port numbers less than `1024` are restricted for privileged users (root) only. If you use a port number that is less than `1024` for an OUD instance, OUD setup, and execution of OUD instance then the commands must be run as privileged user (root). Therefore, you cannot assign these ports to processes running as a regular user. So, if you want to run the server as a regular user, then you use an unprivileged port, for example `1389` by default for LDAP connection. Similarly, you can use the default `1636` port when running as a regular user for SSL-based connection.

This example sets the LDAPS connection handler's listen port to `1636`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
  --set listen-port:1636
```

26.5.1.3 Specifying a Connection Handler's Authorization Policy

Set the `ssl-client-auth-policy` property of the connection handler.

The `ssl-client-auth-policy` property specifies how the connection handler should behave when requesting a client certificate during the SSL or StartTLS negotiation process. If the value is `optional`, the server requests that the client present its own certificate but still accepts the connection even if the client does not provide a certificate. If the value is `required`, the server requests that the client present its own certificate and rejects any connection in which the client does not do so. If the value is `disabled`, the server does not ask the client to present its own certificate.

This example sets the LDAPS connection handler's authorization policy to `required`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
  --set ssl-client-auth-policy:required
```

26.5.1.4 Specifying a Nickname for a Connection Handler's Certificate

Set the `ssl-cert-nickname` property of the connection handler.

The `ssl-cert-nickname` property specifies the nickname of the certificate that the server presents to clients during SSL or StartTLS negotiation. This property is primarily useful when

multiple certificates are in the keystore and you want to specify which certificate is to be used for that listener instance.

This example sets the nickname of the LDAP connection handler's certificate to `server-cert`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set ssl-cert-nickname:server-cert
```

26.5.1.5 Specifying a Connection Handler's Key Manager Provider

Set the `key-manager-provider` property of the connection handler.

The `key-manager-provider` property specifies which key manager provider among the available Configuring Key Manager Providers should be used by the connection handler to obtain the key material for the SSL or StartTLS negotiation.

This example sets the LDAP connection handler's key manager provider to `JKS`. The specified manager must already be configured for the command to succeed.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set key-manager-provider:JKS
```

26.5.1.6 Specifying a Connection Handler's Trust Manager Provider

Set the `trust-manager-provider` property of the connection handler.

The `trust-manager-provider` property specifies which trust manager provider among the available Configuring Trust Manager Providers should be used by the connection handler to decide whether to trust client certificates presented to it.

This example sets the LDAP connection handler's trust manager to `JKS`. The specified manager must already be configured for the command to succeed.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set trust-manager-provider:JKS
```

26.5.1.7 Enabling StartTLS Support

To enable StartTLS support:

1. Specify the appropriate values for the `key-manager-provider` and `trust-manager-provider` properties.
2. Set the `allow-start-tls` property to `true`, as follows:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     set-connection-handler-prop --handler-name "LDAP Connection Handler" \
     --set allow-start-tls:true
   ```

> **Note:**
>
> If SSL is enabled, the `allow-start-tls` property cannot be set.
>
> StartTLS is not supported for connections between the proxy and the remote LDAP servers. Depending on the setting of the remote LDAP server SSL policy, StartTLS client connections can be passed from the proxy to the remote LDAP servers as SSL connections or as insecure connections. For more information, see Creating a Global Index Catalog Containing Global Indexes.

26.5.1.8 Enabling SSL-Based Communication

To enable SSL-based communication:

1. Display the connection handler properties to ensure that the configured key manager provider and trust manager provider values are correct.

   The following example displays the properties of the LDAPS connection handler:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     get-connection-handler-prop --handler-name "LDAPS Connection Handler"
   ```

2. Set the `enabled` property to `true`, as follows:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
     --set enabled:true
   ```

> **Note:**
>
> If SSL is enabled, non-SSL communication will not be available for that connection handler instance.

26.5.1.9 Specifying Protocol Version and Cipher Suites in a Connection Handler

By default, Oracle Unified Directory connection handlers which support SSL/TLS based communication, supports system default SSL/TLS protocols and cipher suites. You can configure `ssl-protocol` and `ssl-cipher-suite` properties of the corresponding connection handlers to override system default SSL/TLS protocols and cipher suites.

See Supported System Default TLS Protocols by Oracle Unified Directory to view the list of TLS protocols and cipher suites supported.
The following connection handlers support the `ssl-protocol` and `ssl-cipher-suite` properties:

- LDAPS
- Admin Connector

The following example enables you to set `ssl-protocol` to TLSv1.1 and the `ssl-cipher-suite` to `TLS_DHE_RSA_WITH_AES_128_CBC_SHA256` for LDAPS Connection Handler:

```
$ dsconfig set-connection-handler-prop \
--handler-name LDAPS Connection Handler \
--set ssl-cipher-suite:TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 \
--set ssl-protocol:TLSv1.1
```

26.5.2 About JMX Connection Handler

The JMX connection handler can be used to communicate with clients using the JMX (Java Management Extensions) protocol. This protocol does not support using StartTLS to allow both encrypted and unencrypted communication over the same port, but you can configure it to accept only unencrypted JMX or only SSL-encrypted JMX communication.

The JMX connection handler provides the server's default configuration for communicating over JMX. To enable SSL for this connection handler, use `dsconfig` to set the following configuration attributes:

Table 26-8 JMX Connection Handler Attributes

Attributes	Description
`key-manager-provider.`	Specifies the DN of the configuration entry for the key manager provider that is used to obtain the key material for the SSL negotiation
`ssl-cert-nickname.`	Specifies the nickname (or alias) of the certificate that is presented to clients
`use-ssl.`	Indicates whether the connection handler is to use SSL to communicate with clients.

The following example uses `dsconfig` in interactive mode to configure the JMX connection handler:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-connection-handler-prop --handler-name "JMX Connection Handler"
```

For a list of the configurable properties, see *JMX Connection Handler Configuration* in *Configuration Reference for Oracle Unified Directory*.

26.6 Configuring SSL Protocol and Cipher Suites in Crypto Manager for Replication

By default, replication between Oracle Unified Directory server instances use SSL-based communication that is facilitated by Crypto Manager. Oracle Unified Directory supports system default protocols and cipher suites to facilitate SSL/TLS communication for replication.

See Supported System Default TLS Protocols by Oracle Unified Directory to view the list of system default TLS protocols and cipher suites supported. This behavior can be overridden by configuring `ssl-protocol` and `ssl-cipher-suite` properties of Crypto Manager. To view the list of configurable properties of Crypto Manager, see “Crypto Manager” in the *Configuration Reference for Oracle Unified Directory*.

The following example enables you to set `ssl-protocol` to `TLSv1.1` and `ssl-cipher-suite` to `TLS_DHE_RSA_WITH_AES_128_CBC_SHA256` of Crypto Manager :

```
$ dsconfig set-crypto-manager-prop \
--set ssl-cipher-suite:TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 \
--set ssl-protocol:TLSv1.1
```

26.7 Overriding System Default Protocols and Cipher Suites for TLS Communication

CLI tools, such as, `ldapsearch` (including other ldap tools), `dsconfig`, `dsreplication`, `ds2oud` uses the system default protocols and cipher suites for TLS communication with Oracle Unified Directory server.

See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to understand about system default values. Follow the procedure provided below to override the settings.

To provide TLS protocol or cipher suites configuration for a particular CLI tool:

1. Create a properties file containing `tls_protocols` and `cipher_suite_sequence` as keys, and desired protocols and cipher suites as their values.
2. Edit the `OUD_INST/OUD/config/java.properties` file, and then add the `custom.config.location` JVM arg that is pointing to the above properties file, in the CLI specific java-args as specified below.

   ```
   ldapsearch.java-args=-client -Dcustom.config.location=/scratch/
   tlsconfig.properties
   ```

3. Run `OUD_INST/OUD/bin/dsjavaproperties` so that the above java arg is in effect.

If you now run any of the CLI tools, for example, `ldapsearch` it would honor the configuration, during any TLS communication with OUD server. Refer to the sample below:

```
tls_protocols=TLSv1.1,TLSv1

cipher_suite_sequence=TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_
WITH_AES_256_CBC_SHA
```

For CLI commands that are outside the OUD instance, or for the commands where Java arguments cannot be configured using INSTANCE_DIR/OUD/config/java.properties, you need to edit the corresponding CLI script, and add the **custom.config.location** java system property specifying the absolute location of the TLS config file.

For example, **`-Dcustom.config.location=/scratch/tlsconfig.properties`**

To configure `oud-replication-gateway-setup/oud-replication-gateway-setup.bat` tool to use specific protocols and cipher suites, you need to edit the script to add the system property. Follow the command given below:

```
"${OPENDS_JAVA_BIN}" -client -Dcustom.config.location=/scratch/
tlsconfig.properties ${SCRIPT_NAME_ARG}
com.sun.gateway.server.tools.setup.ReplicationGatewaySetupLauncher "${@}"
```

For .bat file:

```
"%OPENDS_JAVA_BIN%" -client -Dcustom.config.location=/scratch/
tlsconfig.properties %SCRIPT_NAME_ARG%
com.sun.gateway.server.tools.setup.ReplicationGatewaySetupLauncher %*
```

26.8 Using SASL Authentication

The LDAP protocol definition provides two ways in which clients can authenticate to the server: LDAP simple authentication and SASL authentication.

> **Note:**
>
> SASL is not supported for use with a proxy server instance.

In LDAP simple authentication, the client specifies the DN and password for the user. This is by far the most common authentication mechanism, and in most cases it is also the easiest to use. However, it has several limitations, including the following:

- The user is always required to provide a full DN, rather than something that could be more user-friendly like a user name.
- Only password-based authentication is allowed.
- The client must provide the complete clear-text password to the server.

To address these issues, it is also possible to authenticate clients through the Simple Authentication and Security Layer (SASL), as defined in RFC 4422 (`http://www.ietf.org/rfc/rfc4422.txt`). This is a very extensible framework, and makes it possible for servers to support many different kinds of authentication.

SASL options are described in the following sections:

- About the Supported SASL Mechanisms
- About Authorization IDs
- About the SASL Options for the ANONYMOUS Mechanism
- About the SASL Options for the CRAM-MD5 Mechanism
- About the SASL Options for the DIGEST-MD5 Mechanism
- About the SASL Options for the EXTERNAL Mechanism
- About the SASL Options for the GSSAPI Mechanism
- About the SASL Options for the PLAIN Mechanism
- About DIGEST-MD5 SASL Mechanism

26.8.1 About the Supported SASL Mechanisms

SASL mechanisms are extensible, and the set of information that the client must provide to the server to perform the authentication varies from one mechanism to another.

The following SASL mechanisms are supported:

> **Note:**
>
> With the proxy server, currently the only supported SASL mechanism is ANONYMOUS.

Other supported SASL mechanisms are:

- ANONYMOUS

  This mechanism does not actually authenticate clients, but does provide a mechanis for including trace information in server logs for debugging purposes.

- CRAM-MD5

  This mechanism is provided for backward compatibility only. Do not configure CRAM-MD5 in a production environment. Use the DIGEST-MD5 mechanism instead, because it provides much better security.

- DIGEST-MD5

  This mechanism provides the ability for clients to use password-based authentication without sending the password to the server. Instead, the client only needs to provide information that proves it knows the password. This mechanism offers more options and better security than the CRAM-MD5 mechanism.

- EXTERNAL

  This mechanism provides the ability for clients to identify themselves based on information provided outside of the direct flow of LDAP communication. In Oracle Unified Directory, this may be achieved with SSL client certificates.

- GSSAPI

  This mechanism provides the ability for clients to authenticate to the server through their participation in a Kerberos V5 environment.

- PLAIN

  Support for additional SASL mechanisms can be added by implementing custom SASL mechanism handlers in the server.

Because SASL mechanisms are so extensible, the set of information that the client must provide to the server to perform the authentication varies from one mechanism to another. As such, Oracle Unified Directory clients use a generic interface for users to provide this information. This is exposed through the `-o` or `--saslOption` argument, and the value for this argument should be a name-value pair. Select which SASL mechanism to use using the `mech` option, for example:

```
--saslOption mech=DIGEST-MD5
```

The other options that are available for use depend on the SASL mechanism that has been chosen, as described in the following sections.

26.8.2 About Authorization IDs

Many of the SASL mechanisms below provide the ability to identify a user based on an authorization ID rather than a user DN.

An authorization ID may be given in one of two forms:

- dn:dn

  This is used to provide the full DN of the user to authenticate (for example, dn:uid=john.doe,ou=People,dc=example,dc=com). A value of dn: with no DN is to be treated as the anonymous user, although this form is not accepted by many of the SASL mechanisms listed below.

- u:username

  This is used to provide the username of the user rather than the full DN (for example, u:john.doe).

If the `u:username` form is used, the mechanism that the server uses to resolve that username to the corresponding user entry is based on the identity mapping configuration within the server.

26.8.3 About the SASL Options for the ANONYMOUS Mechanism

The ANONYMOUS mechanism is not really used to perform authentication, no additional options are required.

However, the following option can be supplied:

`trace`

This option can be used to provide a trace string that is written to the server's access log. This can be useful for debugging or to identify the client, although without authentication it is not possible to rely on the validity of this value.

The following command demonstrates the use of SASL anonymous authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=ANONYMOUS \
  --saslOption "trace=Example Trace String" --baseDN "" \
  --searchScope base "(objectClass=*)"
```

26.8.4 About the SASL Options for the CRAM-MD5 Mechanism

The CRAM-MD5 mechanism is used to perform password-based authentication to the server without exposing the clear-text password. It does this by providing an MD5 digest of the clear-text password combined with some randomly-generated data provided by the server, which helps prevent replay attacks.

The SASL CRAM-MD5 mechanism has one SASL option that must be provided:

authid

This specifies the identity of the user that is authenticating to the server. It should be an authorization ID value as described above.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL CRAM-MD5 authentication:

```
ldapsearch --hostname server.example.com --port 1389 --saslOption mech=CRAM-MD5 \
--saslOption authid=u:john.doe --baseDN "" --searchScope base "(objectClass=*)"
```

26.8.5 About the SASL Options for the DIGEST-MD5 Mechanism

The DIGEST-MD5 mechanism is similar to the CRAM-MD5 mechanism, but it is more secure because it combines random data from both the client and the server to help foil both replay and man-in-the-middle attacks.

DIGEST-MD5 authentication also offers several SASL options, including the following:

authid

Specifies the identity of the user that is authenticating to the server. This option must be provided.

realm

This option should not be specified as a DN.

> **Note:**
>
> Do not use the `realm` option, because the server does not use it when mapping identities.

digest-uri

Specifies the digest URI that the client uses to communicate with the server. This is an optional parameter, but if it is provided, specify it in the form `ldap/serveraddress`, where *serveraddress* is the fully-qualified address of the server.

> **Note:**
>
> Do not use the `digest-uri` option in a production environment.

authzid

Specifies the authorization ID that should be used during the authentication process. This option can be used to indicate that the operations requested on the connection after authentication should be performed under the authority of another user.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL DIGEST-MD5 authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=DIGEST-MD5 \
  --saslOption authid=u:john.doe --saslOption realm=dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
```

26.8.6 About the SASL Options for the EXTERNAL Mechanism

The EXTERNAL mechanism is used to perform authentication based on information that is available to the server outside of the LDAP session. At present, this is available only through SSL client authentication, in which case the information that the client's SSL certificate will be used to authenticate that client. As such, it is necessary to use SSL or StartTLS when communicating with the server, and a client certificate keystore must be available.

The EXTERNAL mechanism does not support any additional SASL options. In most cases, it can be requested using either `--saslOption mech=EXTERNAL` or `--useSASLExternal`. The following command demonstrates the use of SASL EXTERNAL authentication:

```
$ ldapsearch --hostname server.example.com --port 1636 --useSSL \
  --keyStorePath /path/to/key.store --keyStorePasswordFile /path/to/key.store.pin \
  --trustStorePath /path/to/trust.store --saslOption mech=EXTERNAL --baseDN "" \
  --searchScope base "(objectClass=*)"
```

For more information, see Configuring SASL External Authentication.

26.8.7 About the SASL Options for the GSSAPI Mechanism

The GSSAPI mechanism is used to perform authentication in a Kerberos V5 environment, and generally requires that the client system be configured to participate in such an environment.

The options available for use with the GSSAPI mechanism include:

- `authid`

  Specifies the authentication ID that should be used to identify the user. This ID should be in the form of a Kerberos principal and not in the authorization ID form described previously. This option must be provided if the user has not authenticated to Kerberos before attempting to bind.

- authzid

  Specifies the authorization ID that should be used to identify the user under whose authority operations should be performed. does not yet support this capability

- `quality-of-protection`

  Specifies the quality of protection to use for the communication. Currently, only the `auth` quality-of-protection value is supported by clients. The `auth-int` and `auth-conf` values are supported by the server.

If the user already has a valid Kerberos ticket on the system when attempting to use GSSAPI, the client attempts to use it so that no password is required. However, if the user does not have a valid Kerberos ticket or if it cannot be accessed for some reason, a password must be provided using either the `--bindPassword` or `--bindPasswordFile` options.

The following command demonstrates the use of SASL GSSAPI authentication for a user that already has a valid Kerberos session:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=GSSAPI \
--saslOption authid=jdoe@EXAMPLE.COM --baseDN "" --searchScope base "(objectClass=*)"
```

26.8.8 About the SASL Options for the PLAIN Mechanism

The PLAIN mechanism provides many of the same capabilities as LDAP simple authentication, although the user may be identified in the form of an authorization ID rather than requiring a full DN.

The following options are available for use when using SASL PLAIN authentication:

- authid

  Specifies the identity of the user that is authenticating to the server. It should be an authorization ID value as described above. This option must be provided.

- authzid

  Specifies the identity of the user under whose authority operations should be performed. It should also be in the form of an authorization ID. does not yet support this capability.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL PLAIN authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=PLAIN \
--saslOption authid=u:john.doe --baseDN "" --searchScope base "(objectClass=*)"
```

26.8.9 About DIGEST-MD5 SASL Mechanism

The DIGEST-MD5 SASL mechanism provides a way for clients to authentication to the Directory Server with a username and password.

The DIGEST-MD5 password does not expose the clear-text password, so it is significantly safer than simple authentication or the PLAIN SASL mechanism when the connection between the client and the server is not secure.

RFC 2831 (`http://www.ietf.org/rfc/rfc2831.txt`) describes the DIGEST-MD5 SASL mechanism, but a revised specification is contained in draft-ietf-sasl-rfc2831bis. The process is as follows:

1. The client sends an message to the server with a bind request protocol op type using an authentication type of `SASL` with a mechanism name of `DIGEST-MD5` and no credentials.
2. The server sends a bind response message back to the client with a result code of 14 (SASL bind in progress) and a server SASL credentials element including, among other things, some randomly-generated data (the nonce).
3. The client takes the nonce provided by the server, and some randomly generated data of its own (the cnonce), an authentication ID, an optional authorization ID, the user's clear-text password, and some other information and uses that to create an MD5 digest. The client then sends a second bind request message including that digest and some other clear-text information back to the server.
4. The server uses the authentication ID to identify the user, and then retrieves the clear-text password for that user (if the clear-text password cannot be obtained, then authentication will fail) and uses it to determine whether the provided digest is valid. The server will then send an appropriate response to the client (usually with a result of either `success` or `invalid credentials`) indicating whether the authentication was successful.
5. If the client requested a quality of protection (QoP) value indicating that the connection should be protected with integrity, confidentiality, or both, then the server will initiate the necessary negotiation with the client. Currently, the directory server does not support the use of the DIGEST-MD5 mechanism with the use of integrity or confidentiality protection.

The DIGEST-MD5 SASL mechanism is very similar to CRAM-MD5 SASL mechanism, but it is somewhat strong because CRAM-MD5 includes only random data from the server whereas DIGEST-MD5 includes random data from both the client and the server. DIGEST-MD5 also provides a provision for ensuring connection integrity, confidentiality, or both that CRAM-MD5 does not offer.

26.9 Configuring SASL Authentication

You can configure directory server to use the various SASL authentication mechanisms.

Some of the SASL authentication mechanisms are:

- Configuring SASL External Authentication
- Configuring SASL DIGEST-MD5 Authentication
- Configuring SASL GSSAPI Authentication

> **Note:**
>
> SASL is not supported for use with a proxy server instance.

26.9.1 Configuring SASL External Authentication

The SASL EXTERNAL mechanism is used to allow a client to authenticate itself to the directory server using information provided outside of what is strictly considered LDAP communication. currently supports authentication using a client certificate presented to the server during SSL or StartTLS negotiation, for LDAP communication only.

The following sections describe the SASL authentication:

- Configuring the LDAP Connection Handler to Allow SASL EXTERNAL Authentication
- Configuring the EXTERNAL SASL Mechanism Handler

26.9.1.1 Configuring the LDAP Connection Handler to Allow SASL EXTERNAL Authentication

For the directory server to be able to map the client certificate to a user entry, ensure that the connection handler is configured to handle client certificates. Use the `dsconfig` to set the following LDAP connection handler properties:

- **ssl-client-auth-policy.** Specifies whether the directory server prompts the client to present its own certificate during the SSL or StartTLS negotiation process. To support SASL EXTERNAL authentication, the value must be either `optional` or `required`. If the value is `disabled`, clients are not prompted to provide a certificate and no certificate is available for authentication.
- **trust-manager-provider.** Specifies the DN of the trust manager provider used to determine whether the directory server trusts the validity of the client certificate. If the server does not trust the client certificate, the SSL or StartTLS negotiation fails and it is not possible for the client to request SASL EXTERNAL authentication. If the server trusts illegitimate client certificates, it is possible for malicious users to forge certificates and impersonate any user in the directory. In most cases, the JKS or PKCS12 trust manager provider should be used and the corresponding trust store loaded only with the issuer certificates that are used to sign client certificates.

> **Note:**
>
> The `dsconfig` command accesses the server configuration over SSL through the administration connector. As such, the relevant connection options must be specified, including how the SSL certificate is trusted. These examples use the `-X` option to trust all certificates.

The following example uses `dsconfig` in interactive mode to set LDAP connection handler properties:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager"-j pwd-file -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler"
```

For a list of the configurable properties, see the "LDAP Connection Handler Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.9.1.2 Configuring the EXTERNAL SASL Mechanism Handler

SASL EXTERNAL bind requests are processed by the SASL mechanism handler. Use the `dsconfig` command to set the following SASL mechanism handler properties:

- **java-class.** Specifies the fully-qualified name of the Java class that provides the logic for the SASL mechanism handler. For the EXTERNAL mechanism, this value is always `org.opends.server.extensions.ExternalSASLMechanismHandler`. An advanced property.
- **enabled.** Indicates whether the EXTERNAL SASL mechanism is enabled for use. If you do not want to allow clients to use SASL EXTERNAL authentication, change its value to `false`.
- **certificate-mapper.** Specifies the DN of the configuration entry for the certificate mapper to be used to map client certificates to user entries.
- **certificate-validation-policy.** Specifies whether the directory server attempts to locate the client certificate in the user's entry after establishing a mapping. If the value is `always`, the authentication succeeds only if the mapped user's entry contains the certificate presented by the client. If the value is `ifpresent` (the default value) and the user's entry contains one or more certificates, the authentication succeeds only if one of those certificates matches the one presented by the client. If the value is `ifpresent` and the user's entry does not contain any certificates, then the authentication still succeeds because it would have been accepted by the trust manager and mapped by the certificate mapper. If the value is `never`, then the server does not attempt to match the certificate to a value in the user's entry even if that entry contains one or more certificates.
- **certificate-attribute.** Specifies the name of the attribute that holds user certificates to be examined if the `certificate-validation-policy` property has a value of either `always` or `ifpresent`.

The following example uses `dsconfig` in interactive mode to set EXTERNAL SASL mechanism handler properties:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager"-j pwd-file -X \
  set-sasl-mechanism-handler-prop --handler-name "EXTERNAL"
```

For a list of the configurable properties, see the "SASL Mechanism Handler Configuration" in the *Configuration Reference for Oracle Unified Directory*.

26.9.2 Configuring SASL DIGEST-MD5 Authentication

The access control and privilege restrictions on a user can be done using the authorization ID keyword (`authzid`). If the user is not using the `authzid` keyword, these restrictions do not apply. Any user that binds using DIGEST-MD5 and the `authzid` keyword must fulfill the following requirements:

- The authentication ID (`authid`) must be granted access by an ACI that grants it the proxy right to the authorization ID.
- The authentication ID (`authid`) entry must contain the `proxied-auth` privilege. The following example creates a test environment and demonstrates the requirements for user authentication using the DIGEST-MD5 SASL mechanism.

The following example creates a test environment and then demonstrates the requirements for a user authentication using the DIGEST-MD5 SASL mechanism.

1. Import the following entries into the directory. These entries define an ACI and three users:

- The entry `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth` privilege but is granted proxy access by the ACI.
- The entry `uid=user.1,ou=People,dc=example,dc=com` has the `proxied-auth` privilege but is not granted proxy access by the ACI.
- The entry `uid=user.2,ou=People,dc=example,dc=com` has the `proxied-auth` privilege and is granted proxy access by the ACI.

```
dn: ou=People,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
objectClass: posixGroup
ou: People
aci: (target="ldap:///uid=proxy user,ou=People,dc=example,dc=com") \
 (targetattr="*") (version 3.0; acl "allow SASL Example"; \
 allow (proxy) userdn="ldap:///uid=user.0,ou=People,dc=example,dc=com ||
 ldap:///uid=user.2,ou=People,dc=example,dc=com";)

dn: uid=user.0,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.0

dn: uid=user.1,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.1
ds-privilege-name: proxied-auth

dn: uid=proxy user,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for proxy user

dn: uid=user.2,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.2
ds-privilege-name: proxied-auth
```

2. Bind using DIGEST-MD5 as `uid=user.1,ou=People,dc=example,dc=com`:

```
$ ldapsearch --port 1389 -j pwd-file --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.1,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
 The SASL DIGEST-MD5 bind attempt failed Result Code: 49 (Invalid Credentials)
```

The search fails because `uid=user.1,ou=People,dc=example,dc=com` is not granted the proxy right by the ACI.

3. Bind using DIGEST-MD5 as `uid=user.0,ou=People,dc=example,dc=com`:

```
$ ldapsearch --port 1389 -j pwd-file --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.0,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed Result Code: 49 (Invalid Credentials)
```

   The search fails because `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth` property.

4. Bind using DIGEST-MD5 as `uid=user.2,ou=People,dc=example,dc=com` authid with both access control access and the `proxied-auth` privilege:

```
$ ldapsearch --port 1389 -j pwd-file --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.2,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
dn:
objectClass: ds-root-dse
objectClass: top
```

   The search succeeds because `uid=user.2,ou=People,dc=example,dc=com` has access allowed by the ACI and the `proxied-auth` privilege.

26.9.3 Configuring SASL GSSAPI Authentication

The access control and privilege restrictions on a user is done by using the authorization ID keyword (`authzid`). If the user is not using the `authzid` keyword, the restrictions do not apply.

Any user that binds using GSSAPI must fulfill the following requirements:

- The authentication ID (`authid`) must be granted access by an ACI that grants it the proxy right to the authorization ID.
- The authentication ID (`authid`) entry must contain the `proxied-auth` privilege.

The following example creates a test environment with three example entries and demonstrates the requirements for user authentication using the GSSAPI SASL mechanism. These examples require a fully configured Kerberos environment, including a valid keytab file.

1. Create three Kerberos principals in the realm `TESTLOCAL.NET`:
   - `user.0@TESTLOCAL.NET`
   - `user.1@TESTLOCAL.NET`
   - `user.2@TESTLOCAL.NET`
2. Configure the GSSAPI SASL handler to be enabled, to use the regular expression identity mapper, and to use a valid TESTLOCAL.NET keytab file.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-sasl-mechanism-handler-prop --handler-name "GSSAPI" \
  --set enabled:true --set identity-mapper:"Regular Expression" \
  --set keytab:keytabPath
```

   The default value of the GSSAPI `enabled` property is `false`, so it must be set to `true`. The default value of `identity-mapper` is Regular Expression. The default value of the `keytab` property is `/etc/krb5/krb5.keytab`.

3. Import the following entries into the directory. These entries define an ACI and three users:

- The entry `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth` privilege but is granted proxy access by the ACI.
- The entry `uid=user.1,ou=People,dc=example,dc=com` has the `proxied-auth` privilege but is not granted proxy access by the ACI.
- The entry `uid=user.2,ou=People,dc=example,dc=com` has the `proxied-auth` privilege and is granted proxy access by the ACI.

```
dn: ou=People,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
objectClass: posixGroup
ou: People
aci: (target="ldap:///uid=proxy user,ou=People,dc=example,dc=com") \
 (targetattr="*") (version 3.0; acl "allow SASL Example"; \
 allow (proxy) userdn="ldap:///uid=user.0,ou=People,dc=example,dc=com"
 || "ldap:///uid=user.2,ou=People,dc=example,dc=com";)

dn: uid=user.0,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=user.0
...
description: This is the description for user.0

dn: uid=user.1,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=user.1
...
description: This is the description for user.1
ds-privilege-name: proxied-auth

dn: uid=user.2,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=user.2
...
description: This is the description for user.2
ds-privilege-name: proxied-auth

dn: uid=proxy user,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=proxy user
...
description: This is the description for proxy user
```

4. Run this command to demonstrate a failing GSSAPI SASL bind using the Kerberos principal, `user.0@TESTLOCAL.NET`:

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.0@TESTLOCAL.NET \
```

```
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed
Result Code:  49 (Invalid Credentials)
```

This search fails because `user.0@TESTLOCAL.NET` maps to `uid=user.0,ou=People,dc=example,dc=com`, which has access control permissions to `uid=proxy user,ou=People,dc=example,dc=com` but does not have the `proxied-auth` privilege.

5. Run this command to demonstrate a failing GSSAPI SASL bind using the Kerberos principal, `user.1@TESTLOCAL.NET`.

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.1@TESTLOCAL.NET \
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed
Result Code:  49 (Invalid Credentials)
```

This search fails because `user.1@TESTLOCAL.NET` maps to `uid=user.1,ou=People,dc=example,dc=com`, which has the `proxied-auth` privilege but does not have access control permissions to `uid=proxy user,ou=People,dc=example,dc=com`.

6. Run this command to demonstrate a successful GSSAPI SASL bind using the Kerberos principal `user.2@TESTLOCAL.NET`:

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.2@TESTLOCAL.NET \
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
dn:
objectClass: ds-root-dse
objectClass: top }}} \\ \\
```

This search succeeds because `user.2@TESTLOCAL.NET` maps to `uid=user.2,ou=People,dc=example,dc=com`, which has both the `proxied-auth` privilege and access control permission to `id=proxy user,ou=People,dc=example,dc=com`.

26.10 Configuring Kerberos and the Oracle Unified Directory Server for GSSAPI SASL Authentication

You can configure and Kerberos Version 5 for GSSAPI SASL authentication.

The following sections describe configuring Kerberos and Oracle Unified Directory:

- Configuring Kerberos V5 on a Host
- Specifying SASL Options for Kerberos Authentication
- Configuring Kerberos Authentication Using GSSAPI With SASL
- Troubleshooting Kerberos Configuration

26.10.1 Configuring Kerberos V5 on a Host

You can configure Kerberos V5 on the host machine where your LDAP clients will run.

Perform the following steps to configure Kerberos V5 on a host:

1. Install Kerberos V5 according to its installation instructions.

> **Note:**
>
> Previously, you were advised to install the Sun Enterprise Authentication Mechanism 1.0.1 client software.
>
> Starting with the Oracle Solaris 10 release, the necessary Sun Enterprise Authentication Mechanism 1.0.1 client software components were incorporated into Solaris. If you are using Oracle Solaris release 10 or later, installing that client software is no longer necessary.

2. Configure the Kerberos software.

   For Solaris: Using the Sun Enterprise Authentication Mechanism software, configure the files under `/etc/krb5`. This configuration sets up the `kdc` server, and defines the default realm and any other configuration required by your Kerberos system.

3. If necessary, modify the file `/etc/gss/mech` so that the first value that is listed is `kerberos_v5`.

26.10.2 Specifying SASL Options for Kerberos Authentication

You can specify appropriate SASL options for the Kerberos installation.

Perform the following steps for Kerberos installation:

1. Before using a client application that is enabled with the GSSAPI mechanism, initialize the Kerberos security system with your user Principal.

   ```
   $ kinit user-principal
   ```

   where the *user-principal* is your SASL identity, for example, `bjensen@example.com`.

2. Specify SASL options for using Kerberos.

   In the UNIX environment, you must set the `SASL_PATH` environment variable to the correct path for the SASL libraries. For example in the Korn shell:

   ```
   $ export SASL_PATH=SASL-library
   ```

   This path assumes that the Oracle Unified Directory software is installed on the same host where the LDAP tools are invoked.

   The following example of the `ldapsearch` tool shows the use of the `-o` (lowercase letter o) option to specify SASL options for using Kerberos:

   ```
   $ ldapsearch -h www.host1.com -p 1389 -o mech=GSSAPI -o authid="bjensen@EXAMPLE.COM"
   \
    -o authzid="bjensen@EXAMPLE.COM" -b "dc=example,dc=com" "(givenname=Richard)"
   ```

The `authid` can be omitted because it is present in the Kerberos cache that was initialized by the `kinit` command. If `authid` is present, `authid` and `authzid` must be identical, although the `authzid` intended for proxy operations is not used. The value of `authid` is the Principal that is used in identity mapping. The Principal must be the full Principal, including the realm.

26.10.3 Configuring Kerberos Authentication Using GSSAPI With SASL

Configuring Kerberos for the Oracle Unified Directory directory server can be complicated. Your first point of reference should be the Kerberos documentation.

For more help, use the following example procedure to get an idea of which steps to follow. Be aware, however, that this procedure is an example. You must modify the procedure to suit your own configuration and your own environment.

Additional information about configuring and using Kerberos in the Solaris OS can be found in *System Administration Guide: Security Services*. This guide is a part of the Solaris documentation set. You can also consult the man pages.

Information about this example and the steps used are as follows:

1. Assumptions for This Example
2. Editing the Kerberos Client Configuration File(All machines)
3. Editing the Administration Server ACL Configuration File(All machines)
4. Editing the KDC Server Configuration File (KDC Machine)
5. Creating the KDC Database (KDC Machine)
6. Creating an Administration Principal and Keytab(KDC Machine)
7. Start the Kerberos Daemons(KDC Machine)
8. Adding Host Principals for the KDC and Oracle Unified Directory Machines(KDC Machine)
9. Adding an LDAP Principal for the Directory Server(KDC Machine)
10. Adding a Test User to the KDC(KDC Machine)
11. Directory Server Machine: Install Oracle Unified Directory
12. Configuring the Directory Server to Enable GSSAPI(Directory Server Machine)
13. Creating and Configuring the Directory Server LDAP(Directory Server Machine)
14. Adding a Test User to the Directory Server(Directory Server Machine)
15. Obtaining a Kerberos Ticket as the Test User(Directory Server Machine)
16. Authenticating to the Directory Server Through GSSAPI(Client Machine)

26.10.3.1 Assumptions for This Example

This example procedure describes the process of configuring one machine to operate as a Key Distribution Center (KDC), and a second machine to run the directory server. The result of this procedure is that users can perform Kerberos authentication through GSSAPI.

It is possible to run both the KDC and the directory server on the same machine. If you choose to run both on the same machine, use the same procedure, but omit the steps for the directory server machine that have already been done for the KDC machine.

This procedure makes several assumptions about the environment that is used. When using the example procedure, modify the values accordingly to suit your environment. These assumptions are:

- This system has a fresh installation of the Solaris 10 software with the latest recommended patch cluster installed. Kerberos authentication to the directory server can fail if the appropriate Solaris patches are not installed.
- The machine that is running the Kerberos daemons has the fully qualified domain name of `kdc.example.com`. The machine must be configured to use DNS as a naming service. This configuration is a requirement of Kerberos. Certain operations might fail if other naming services such as `file` are used instead.
- The machine that is running the directory server has the fully qualified domain name of `directory.example.com`. This machine must also be configured to use DNS as a naming service.
- The directory server machine serves as the client system for authenticating to the directory server through Kerberos. This authentication can be performed from any system that can communicate with both the directory server and Kerberos daemons. However, all of the necessary components for this example are provided with the Oracle Unified Directory directory server, and the authentication is performed from that system.
- Users in the directory server have DNs of the form `uid=`*username*`,ou=People,dc=example,dc=com`. The corresponding Kerberos principal is *username*`@EXAMPLE.COM`. If a different naming scheme is used, a different GSSAPI identity mapping must be used.

26.10.3.2 Editing the Kerberos Client Configuration File(All machines)

The `/etc/krb5/krb5.conf` configuration file provides information that Kerberos clients require to communicate with the KDC.

Edit the `/etc/krb5/krb5.conf` configuration file on the KDC machine, the directory server machine, and any client machines that will authenticate to the directory server using Kerberos.

- Replace every occurrence of "`___default_realm___`" with "`EXAMPLE.COM`".
- Replace every occurrence of "`___master_kdc___`" with "`kdc.example.com`".
- Remove the lines that contain "`___slave_kdcs___`" as there will be only a single Kerberos server.
- Replace "`___domain_mapping___`" with "`.example.com = EXAMPLE.COM`" (note the initial period in`.example.com`).

The updated `/etc/krb5/krb5.conf` configuration file should look like the contents of the following example.

26.10.3.2.1 Edited Kerberos Client Configuration File */etc/krb5/krb5.conf*

```
#pragma ident   "@(#)krb5.conf  1.2     99/07/20 SMI"
# Copyright (c) 1999, by Sun Microsystems, Inc.
# All rights reserved.
#
# krb5.conf template
# In order to complete this configuration file
# you will need to replace the __<name\>__ placeholders
# with appropriate values for your network.
#
```

```
[libdefaults]
        default_realm = EXAMPLE.COM
[realms]
        EXAMPLE.COM = {
                kdc = kdc.example.com
                admin_server = kdc.example.com
        }
[domain_realm]
        .example.com = EXAMPLE.COM
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log
        kdc_rotate = {

# How often to rotate kdc.log. Logs will get rotated no more
# often than the period, and less often if the KDC is not used
# frequently.
                period = 1d

# how many versions of kdc.log to keep around (kdc.log.0, kdc.log.1, ...)
                versions = 10
        }

[appdefaults]
        kinit = {
                renewable = true
                forwardable= true
        }
        gkadmin = {
                help_url =
 http://docs.sun.com:80/ab2/coll.384.1/SEAM/@AB2PageView/1195
        }
```

26.10.3.3 Editing the Administration Server ACL Configuration File(All machines)

Replace "___default_realm___" with "EXAMPLE.COM" in the /etc/krb5/kadm5.acl configuration file. The updated file should look like the following example.

Edited Administration Server ACL Configuration File

```
#
# Copyright (c) 1998-2000 by Sun Microsystems, Inc.
# All rights reserved.
#
# pragma ident   "@(#)kadm5.acl  1.1     01/03/19 SMI"
*/admin@EXAMPLE.COM *
```

26.10.3.4 Editing the KDC Server Configuration File (KDC Machine)

Edit the /etc/krb5/kdc.conf file to replace "___default_realm___" with "EXAMPLE.COM". The updated file should look like the following example.

Edited KDC Server Configuration File /etc/krb5/kdc.conf

```
# Copyright 1998-2002 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "@(#)kdc.conf   1.2     02/02/14 SMI"

[kdcdefaults]
```

```
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                admin_keytab = /etc/krb5/kadm5.keytab
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                default_principal_flags = +preauth
        }
```

26.10.3.5 Creating the KDC Database (KDC Machine)

```
$ /usr/sbin/kdb5_util create -r EXAMPLE.COM -s
Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key: password
Re-enter KDC database master key to verify: password
$
```

26.10.3.6 Creating an Administration Principal and Keytab(KDC Machine)

Use the following command to create an administration user with a Principal of `kws/admin@EXAMPLE.COM` and service keys that will be used by the administration daemon.

```
$ /usr/sbin/kadmin.local
kadmin.local:  add_principal kws/admin
Enter password for principal "kws/admin@EXAMPLE.COM": secret
Re-enter password for principal "kws/admin@EXAMPLE.COM": secret
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/kdc.example.com
Entry for principal kadmin/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab changepw/kdc.example.com

Entry for principal changepw/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/changepw
Entry for principal kadmin/changepw with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  quit$
```

26.10.3.7 Start the Kerberos Daemons(KDC Machine)

The Kerberos daemons are managed by the Service Management Facility (SMF) framework. Run the following commands to start the KDC and administration daemons:

```
$ /etc/init.d/kdc start
$ /etc/init.d/kdc.master start
$

$ svcadm disable network/security/krb5kdc
$ svcadm enable network/security/krb5kdc
$ svcadm disable network/security/kadmin
```

```
$ svcadm enable network/security/kadmin
$
```

The KDC process appears in the process list as `/usr/lib/krb5/krb5kdc`. The administration daemon appears as `/usr/lib/krb5/kadmind`.

26.10.3.8 Adding Host Principals for the KDC and Oracle Unified Directory Machines(KDC Machine)

Use the following sequence of commands to add host Principals to the Kerberos database for the KDC and the directory server machines. The host Principal is used by certain Kerberos utilities such as `klist`.

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  add_principal -randkey host/kdc.example.com
Principal "host/kdc.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/kdc.example.com
Entry for principal host/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  add_principal -randkey host/directory.example.com
Principal "host/directory.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/directory.example.com
Entry for principal host/directory.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  quit
$
```

26.10.3.9 Adding an LDAP Principal for the Directory Server(KDC Machine)

For the directory server to be able to validate the Kerberos tickets that are held by authenticating users, the directory server must have its own Principal. Currently Oracle Unified Directory is hard coded to require a Principal of `ldap/fqdn@realm` where *fqdn* is the fully-qualified domain name of the directory server and *realm* is the Kerberos realm. The *fqdn* must match the fully qualified name that is provided when you install Oracle Unified Directory. In this case, the Principal for the directory server would be `ldap/directory.example.com@EXAMPLE.COM`.

Use the following sequence of commands to create an LDAP Principal for the directory server:

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin: add_principal -randkey ldap/directory.example.com
Principal "ldap/directory.example.com@EXAMPLE.COM" created.
kadmin: quit
$
```

26.10.3.10 Adding a Test User to the KDC(KDC Machine)

To perform Kerberos authentication, the user authenticating must exist in the Kerberos database. In this example, the user has the user name `kerberos-test`, which means that the Kerberos Principal is `kerberos-test@EXAMPLE.COM`.

Create the user by using the command sequence in this example:

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
```

```
kadmin:  add_principal kerberos-test
Enter password for principal "kerberos-test@EXAMPLE.COM": secret

Re-enter password for principal "kerberos-test@EXAMPLE.COM": secret

Principal "kerberos-test@EXAMPLE.COM" created.
kadmin:  quit
$
```

26.10.3.11 Directory Server Machine: Install Oracle Unified Directory

Install Oracle Unified Directory. The following table lists the installation settings that this section uses in examples.

Variable Type	Example Value
Fully qualified directory server DNS name	`directory.example.com`
Server port	`389`
Suffix	`dc=example,dc=com`
Installation directory	`/asinst_1/oud`
Oracle Unified Directory server user	`oud`
Oracle Unified Directory server group	`oud`
Kerberos test principal	`kerberos-test`
Oracle Unified Directory keytab path	`/asinst_1/oud/config/oud.keytab`

> **Note:**
>
> The fully qualified directory server DNS name must resolve to the same IP address on all of the servers (the Oracle Unified Directory servers and the Kerberos Key Distribution Center (KDC) and client machines that expect to bind to the server using GSSAPI SASL).

26.10.3.12 Creating and Configuring the Directory Server LDAP(Directory Server Machine)

As mentioned previously, to authenticate Kerberos users through GSSAPI, Oracle Unified Directory must have its own Principal in the KDC. The Principal information must reside in a Kerberos keytab on the directory server machine. This information must be in a file that is readable by the user account under which the directory server operates.

> **Note:**
>
> This step must be performed before the `GSSAPI SASL mechanism handler` is configured. The handler checks to make sure the keytab file exists before it will initialize.

Create a keytab file with the correct properties by using the following command sequence:

```
$ kadmin -p kws/admin@EXAMPLE.COM
kadmin:  addprinc -randkey ldap/directory.example.com
WARNING: no policy specified for ldap/directory.example.com@EXAMPLE.COM;
  defaulting to no policy
Principal "ldap/directory.example.com@EXAMPLE.COM" created.
kadmin:  ktadd -k asinst_1/oud/config/oud.keytab ldap/directory.example.com
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type AES-128 CTS mode
  with 96-bit SHA-1 HMAC added to keytab WRFILE:asinst_1/oud/config/oud.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type Triple DES cbc mode
  with HMAC/sha1 added to keytab WRFILE:asinst_1/oud/config/oud.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type ArcFour with HMAC/md5
  added to keytab WRFILE:asinst_1/oud/config/oud.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type DES cbc mode with RSA-MD5
  added to keytab WRFILE:asinst_1/oud/config/oud.keytab.
kadmin:  quit
```

Change the permissions and ownership on this custom keytab. Make the keytab owned by the user account used to run the directory server and readable only by that user:

```
$ chown oud:oud asinst_1/oud/config/oud.keytab
$ chmod 600 asinst_1/oud/config/oud.keytab
```

To allow these changes to take effect, stop and restart the directory server.

26.10.3.13 Configuring the Directory Server to Enable GSSAPI(Directory Server Machine)

This step shows examples of managing the `GSSAPI SASL mechanism handler` on the directory server host `directory.example.com`.

Use the `dsconfig` command as shown in the following example to enable the GSSAPI SASL mechanism handler on the directory server host `directory.example.com` and configure it to use the `asinst_1/oud/config/oud.keytab`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -j pwd-file
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:true \
  --set keytab:asinst_1/oud/config/oud.keytab \
  --set server-fqdn:directory.example.com
```

The last line in this command sets the GSSAPI SASL mechanism property `server-fqdn` to `directory.example.com`. This is an optional parameter, which can be left out only if it is assured that a hostname lookup on the directory server host returns the exact hostname that was used in creating the LDAP principal. Setting this property explicitly assures that the two names are the same (in this example, `directory.example.com`).

Confirm that the configuration is correct by examining the properties of the GSSAPI SASL mechanism handler on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -j pwd-file \
  get-sasl-mechanism-handler-prop \
  --handler-name GSSAPI
```

```
Property              : Value(s)
----------------------:----------------------
enabled               : true
identity-mapper       : Regular Expression
kdc-address           : -
keytab                : asinst_1/oud/config/oud.keytab
principal-name        : -
quality-of-protection : none
realm                 : -
server-fqdn           : directory.example.com
```

If necessary for troubleshooting, you can use `dsconfig` to list the status of all the SASL mechanism handlers on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -j pwd-file \
  list-sasl-mechanism-handlers
SASL Mechanism Handler : Type       : enabled
----------------------:------------:--------
ANONYMOUS              : anonymous  : false
CRAM-MD5               : cram-md5   : true
DIGEST-MD5             : digest-md5 : true
EXTERNAL               : external   : true
GSSAPI                 : gssapi     : true
PLAIN                  : plain      : true
```

If necessary, you can use `dsconfig` to disable the GSSAPI SASL mechanism handler on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -j pwd-file \
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:false
```

26.10.3.14 Adding a Test User to the Directory Server(Directory Server Machine)

To authenticate a Kerberos user to the directory server, there must be a directory entry for the user that corresponds to the Kerberos Principal for that user.

In a previous step, a test user was added to the Kerberos database with a Principal of `kerberos-test@EXAMPLE.COM`. Because of the identity mapping configuration added to the directory, the corresponding directory entry for that user must have a DN of `uid=kerberos-test,ou=People,dc=example,dc=com`.

Before you can add the user to the directory, you must create the file `testuser.ldif` with the following contents.

```
dn: uid=kerberos-test,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: kerberos-test
givenName: Kerberos
sn: Test
cn: Kerberos Test
description: An account for testing Kerberos authentication through GSSAPI
```

Next, use `ldapmodify` to add this entry to the server:

```
$ ldapmodify -D "cn=Directory Manager" -w - -f testuser.ldif
adding new entry uid=kerberos-test,ou=People,dc=example,dc=com
$
```

26.10.3.15 Obtaining a Kerberos Ticket as the Test User(Directory Server Machine)

The test user exists in the Kerberos database, the directory server, and the KDC. Therefore, it is now possible to authenticate as the test user to the directory server over Kerberos through GSSAPI.

First, use the `kinit` command to get a Kerberos ticket for the user, as shown in the following example:

```
$ kinit kerberos-test
Password for kerberos-test@EXAMPLE.COM: secret
$
```

Then, use the `klist` command to view information about this ticket:

```
$ klist
Kerberos 5 ticket cache: 'API:6'
Default principal: kerberos-test@EXAMPLE.COM
Valid Starting     Expires            Service Principal
03/23/09 12:35:05  03/23/09 20:35:05  krbtgt/EXAMPLE.COM@EXAMPLE.COM
renew until 03/30/09 12:34:15
$
```

26.10.3.16 Authenticating to the Directory Server Through GSSAPI(Client Machine)

The final step is to authenticate to the directory server by using GSSAPI. The `ldapsearch` utility provided with The directory server provides support for SASL authentication, including GSSAPI, DIGEST-MD5, and EXTERNAL mechanisms. However, to bind by using GSSAPI you must provide the client with the path to the SASL library. Provide the path by setting the `SASL_PATH` environment variable to the `lib/sasl` directory:

```
$ SASL_PATH=SASL-library
$ export SASL_PATH
$
```

To actually perform a Kerberos-based authentication to the directory server using `ldapsearch`, you must include the `-o mech=GSSAPI` and `-o authzid=`*principal* arguments.

You must also specify the fully qualified host name, shown here as `-h directory.example.com`, which must match the value of the `nsslapd-localhost` attribute on `cn=config` for the server. This use of the `-h` option is needed because the GSSAPI authentication process requires the host name provided by the client to match the host name provided by the server.

The following example retrieves the `dc=example,dc=com` entry while authenticated as the Kerberos test user account created previously:

```
$ldapsearch -h directory.example.com -p 389 -o mech=GSSAPI \
-o authzid="kerberos-test@EXAMPLE.COM"
-b "dc=example,dc=com" -s base "(objectClass=*)"
version: 1
dn: dc=example,dc=com
dc: example
objectClass: top
```

```
objectClass: domain
$
```

Check the directory server access log to confirm that the authentication was processed as expected:

```
$ tail -12 /local/ds/logs/access

[24/Jul/2004:00:30:47 -0500] conn=0 op=-1 msgId=-1 - fd=23 slot=23 LDAP
        connection from 1.1.1.8 to 1.1.1.8
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - RESULT err=0 tag=97
     nentries=0 etime=0 dn="uid=kerberos-test,ou=people,dc=example,dc=com"
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - SRCH base="dc=example,dc=com"
     scope=0 filter="(objectClass=*)" attrs=ALL
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - RESULT err=0 tag=101 nentries=1
     etime=0
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=5 - UNBIND
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=-1 - closing - U1
[24/Jul/2004:00:30:48 -0500] conn=0 op=-1 msgId=-1 - closed.
$
```

This example shows that the bind is a three-step process. The first two steps return LDAP result `14` (SASL bind in progress), and the third step shows that the bind was successful. The `method=sasl` and `mech=GSSAPI` tags show that the bind used the GSSAPI SASL mechanism. The `dn="uid=kerberos-test,ou=people,dc=example,dc=com"` at the end of the successful bind response shows that the bind was performed as the appropriate user.

26.10.4 Creating a Kerberos Workflow Element Using `dsconfig`

You can create a Kerberos workflow element by running the `dsconfig create-workflow-element` command.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
 create-workflow-element \
 --type KerberosAuthProviderWorkflowElement  \
 --element-name Kerberos_Test_WE  \
```

26.10.5 Troubleshooting Kerberos Configuration

You can check the conditions to troubleshoot Kerberos configuration.

If the Kerberos installation does not perform as expected, check the following conditions:

- Perform a successful `kinit` using the test principal from the directory server machine to ensure that the directory server can authenticate to the Kerberos KDC.
- Perform a successful `kinit` using the test principal from the client machines to ensure that the client machines can authenticate to the Kerberos KDC.

- Ensure that the directory server's keytab file exists and is readable by the directory server. That is, ensure that the keytab file's ownership and permission settings are correct.
- Ensure that the LDAP principal name in the keytab file matches the hostname that the directory server used when it was configured. The following example shows a configuration that fails:

  1. Configure GSSAPI as shown below. The value specified for the `server-fqdn` attribute, `bad.example.com`, does not match the value used in creating the keytab, `directory.example.com`.

     ```
     $ dsconfig -X -n -p 4444 -h directory.example.com \
       -D "cn=directory manager" -j pwd-file \
       set-sasl-mechanism-handler-prop \
       --handler-name GSSAPI \
       --set enabled:true \
       --set keytab:asinst_1/oud/config/oud.keytab \
       --set server-fqdn:bad.example.com
     ```

  2. From a client, attempt an ldapsearch authenticating using GSSAPI.

     ```
     $ ldapsearch -h directory.example.com \
       -o mech=GSSAPI -o authid=kerberos-test@EXAMPLE.COM \
       --searchScope base \
       -b "uid=kerberos-test,ou=people,dc=example,dc=com" "(objectclass=*)"
     An error occurred while attempting to perform GSSAPI authentication to
       the Directory Server: \
     PrivilegedActionException(AccessController.java:-2)
     Result Code:  82 (Local Error)
     ```

     The search fails as expected.

  3. To determine the cause of the search failure, inspect the directory server's access log:

     ```
     $ tail asinst_1/oud/logs/access
     [23/Mar/2009:13:12:59 -0500] CONNECT conn=14 from=129.150.33.77:65076
       to=192.168.0.199:1389 protocol=LDAP
     [23/Mar/2009:13:13:00 -0500] BIND REQ conn=14 op=0 msgID=1
       type=SASL mechanism=GSSAPI dn=""
     [23/Mar/2009:13:13:00 -0500] BIND RES conn=14 op=0 msgID=1
       result=49 authFailureID=1310915   authFailureReason="An unexpected error
       occurred while trying to create an GSSAPI context:
       major code (13) No valid credentials provided,
       minor code (-1)   Failed to find any Kerberos Key" etime=253
     [23/Mar/2009:13:13:00 -0500] DISCONNECT conn=14 reason="Client Disconnect"
     ```

     The message in the minor code of the last record in the access log shows that the directory server could not find a match in the keytab file.

  4. To fix the situation, disable the handler and then re-enable it with the correct information, as shown in the following example.

     ```
     $ dsconfig -X -n -p 4444 -h directory.example.com \
       -D "cn=directory manager" -j pwd-file \
       set-sasl-mechanism-handler-prop \
       --handler-name GSSAPI \
       --set enabled:false
     $ dsconfig -X -n -p 4444 -h directory.example.com
       -D "cn=directory manager" -j pwd-file \
       set-sasl-mechanism-handler-prop \
       --handler-name GSSAPI \
       --set enabled:true \
       --set keytab:asinst_1/oud/config/oud.keytab \
       --set server-fqdn:directory.example.com
     ```

```
$ ldapsearch -h directory.example.com \
  -o mech=GSSAPI \
  -o authid=kerberos-test@EXAMPLE.COM \
  --searchScope base \
  -b "uid=kerberos-test,ou=people,dc=example,dc=com" "(objectclass=*)"
dn: uid=kerberos-test,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: kerberos-test
givenName: Kerberos
sn: Test
cn: Kerberos Test
description: An account for testing Kerberos authentication through GSSAPI
```

26.11 Testing SSL, StartTLS, and SASL Authentication With `ldapsearch`

The `ldapsearch` utility included with the directory server is useful for testing that the server is properly configured to support SSL and StartTLS.

This utility includes several options that are well-suited for testing in various scenarios. This section describes how to use `ldapsearch` to test SSL and StartTLS communication, and SASL EXTERNAL authentication. The same process can be used with many of the other client tools provided with the directory server, including `ldapmodify`, `ldapcompare`, and `ldapdelete`. Detailed information is described in the following sections:

- ldapsearch Command Line Arguments Applicable To Security
- Testing SSL
- Testing StartTLS
- Testing SASL External Authentication

26.11.1 ldapsearch Command Line Arguments Applicable To Security

You can use the command-line arguments when using the `ldapsearch` tool to communicate through SSL or StartTLS:

Table 26-9 ldapsearch Command Line Arguments

Arguments	Description
`-h address` or `--hostname address`	Specifies the address of the directory server to which you want to connect. If no value is specified, the IPv4 loopback address (`127.0.0.1`) is used.
`-p port` or `--port port`	Specifies the port number on which the directory server is listening for connections. If no value is specified, the standard unencrypted LDAP port (`389`) is used
`-Z` or `--useSSL`	Indicates that the client should use SSL to secure communication with the directory server. If this option is used, the value specified for the port argument must be one on which the server is listening for SSL-based connections. The default LDAPS port is `636`

Table 26-9 (Cont.) ldapsearch Command Line Arguments

Arguments	Description
`-q` or `--useStartTLS`	Indicates that the client should use the `--useStartTLS` extended operation to secure communication with the directory server. If this option is used, the value specified for the port argument must be the one on which the server is listening for clear-text LDAP connections. The port argument is not required if the server is listening on the default LDAP port (`389`).
`-r` or `--useSASLExternal`	Indicates that the client should use SASL EXTERNAL authentication to authenticate to the directory server. If this option is used, you must also provide a keystore path.
`-X` or `--trustAll`	Indicates that the client should blindly trust any certificate that the directory server presents. Do not use this option with the argument used to specify the trust store path.
`-K path` or `--keyStorePath path`	Specifies the path to the keystore that should be used if the client is to present a certificate to the directory server (for example, when using SASL EXTERNAL authentication). This should be the path to a JKS keystore.
`-W password` or `--keyStorePassword password`	Specifies the PIN required to access the contents of the key tore. Do not use this option with the keystore password file argument.
`--keyStorePasswordFile path`	Specifies the path to a file containing the PIN required to access the contents of the keystore. Do not use this option with the keystore password argument.
`-N nickname` or `--certNickname nickname`	Specifies the nickname, or alias, of the certificate that the client should present to the directory server. The keystore path argument must also be provided. If no nickname is given, then the client will pick the first acceptable client certificate that it finds in the keystore.
`-P path` or `--trustStorePath path`	Specifies the path to the JKS trust store file that the client should use when determining whether to trust the certificate presented by the directory server. If this argument is not given and the `trustAll` option is not given, then any certificate presented to the client will be displayed and the user will be prompted about whether to trust it.
`--trustStorePassword password`	Specifies the password needed to access the trust store contents. In most cases, no trust store password is required. Do not use this option with the trust store password file option.
`--trustStorePasswordFile path`	Specifies the path to a file containing the password needed to access the trust store contents. In most cases, no trust store password is required. Do not use this option with the trust store password option.
`-E` or `--reportAuthzID`	Indicates that the directory server should include the authorization identity of the authenticated user in the bind response. This is useful when performing SASL authentication to determine the user to which the client certificate (or other form of SASL credentials if a mechanism other than EXTERNAL was used) was mapped.

26.11.2 Testing SSL

You can use `ldapsearch` to communicate with a directory server using LDAP over SSL.

The following demonstrates the use of `ldapsearch`:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --baseDN "" --searchScope base "(objectClass=*)"
```

In this case, no trust store was specified, and the `--trustAll` argument was also not given. Therefore, when the server presents its certificate to the client, the user will be prompted about whether that certificate should be trusted. The entire sequence might look something like:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --baseDN "" --searchScope base "(objectClass=*)"

The server is using the following certificate:
Subject DN: CN=directory.example.com, O=Example Corp, C=US
Issuer DN: CN=directory.example.com, O=Example Corp, C=US
Validity: Fri Mar 02 16:48:17 CST 2007 through Thu might 31 17:48:17 CDT 2007
Do you want to trust this certificate and continue connecting to the server?
Please enter "yes" or "no":
dn:
objectClass: ds-rootDSE
objectClass: top
```

If the client simply wants to always trust any certificate that the server presents without being prompted, then the `--trustAll` argument might be provided. For example:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --trustAll --baseDN "" --searchScope base \
"(objectClass=*)"
```

If the client has a trust store and wants to use that to determine whether to trust the server certificate, then the `--trustStorePath` argument might also be given. For example:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --trustStorePath client.truststore --baseDN "" \
--searchScope base "(objectClass=*)"
```

26.11.3 Testing StartTLS

The process for using StartTLS with the `ldapsearch` utility is almost identical to the process for using SSL. The only differences are that you should use the port on which the server is listening for unencrypted LDAP requests and that you should indicate that StartTLS should be used instead of SSL (that is, use `--useStartTLS` instead of `--useSSL`).

The following example is the equivalent of the first example given for using SSL with `ldapsearch` except that it uses StartTLS to secure the communication:

```
$ ldapsearch -h directory.example.com --port 1389 \
--useStartTLS --baseDN "" --searchScope base "(objectClass=*)"
```

This applies to all of the other examples given. Simply change the port number from the LDAPS port to the LDAP port, and replace the `--useSSL` option with `--useStartTLS`.

26.11.4 Testing SASL External Authentication

SASL EXTERNAL authentication might be used with either SSL or StartTLS. The primary differences are that it will be necessary to provide a keystore that contains the client certificate,

the PIN required to access the contents of that keystore, and a flag indicating that the client should use SASL EXTERNAL authentication.

> **Note:**
>
> SASL is not supported for use with a proxy server instance.

The following example demonstrates sample usage for such a command:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --keyStorePath /path/to/client.keystore \
--keyStorePasswordFile /path/to/client.keystore.pin \
--useSASLExternal --certNickName nickname \
--baseDN "" --searchScope base \
"(objectClass=*)"
```

When using SASL EXTERNAL authentication, it is also often useful to ask the server to return the authorization identity to ensure that the authentication is being performed as the correct user. The following demonstrates an example of this process. (Note the value reported on the line beginning with the "#" character.)

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --keyStorePath /path/to/client.keystore \
--keyStorePasswordFile /path/to/client.keystore.pin \
--useSASLExternal --reportAuthzID --certNickName nickname \
--baseDN "" --searchScope base "(objectClass=*)"

# Bound with authorization ID dn:uid=test.user,dc=example,dc=com
dn:
objectClass: ds-rootDSE
objectClass: top
```

26.12 Debugging SSL Using OpenSSL `s_client` Test Utility

OpenSSL provides an extremely valuable and useful diagnostic tool, called `s_client`, to debug SSL servers.

These topics describe the OpenSSL `s_client` test utility and the solutions to debug different scenarios:

- About OpenSSL `s_client` Test Utility
- Scenario 1- Connection Refused
- Scenario 2- Verify Return Code: 18 (Self Signed Certificate)
- Scenario 3 - Verify Return Code: 0 (ok)
- Scenario 4 - SSLHandshakeException
- Scenario 5 - SASL EXTERNAL Bind Request Could Not Be Processed

26.12.1 About OpenSSL `s_client` Test Utility

`s_client` is a diagnostic tool used to debug SSL servers. The command implements a generic SSL/TLS client which connects to a remote host using SSL/TLS.

This utility lets you test or debug servers that use SSL/TLS with a powerful command line utility. To test the secure connections to the Oracle Unified Directory server, type the following command on the command prompt:

```
openssl s_client -connect <host>:<port> [options]
```

Here:

`s_client`: It is an SSL/TLS test client, which is used to test secure servers. The test client can connect to a secure port, while providing a detailed log of the steps performed during the SSL/TLS handshake.

`hostname:port`: This specifies the host and optional port to connect to. If not specified then an attempt is made to connect to the local host on port `443`, because https uses port `443`.

If connected, you can manually type in several commands, such as "**`GET /`**" and "**`HEAD / HTTP/1.0`**" for secure servers. However, if the handshake fails then there are several possible causes. If you want to know the problem you are experiencing is related to the application, firewall, certificate trust, or so on then this section describes a way to eliminate SSL from your list of usual suspects.

26.12.2 Scenario 1- Connection Refused

You connect the SSL client over the designated SSL port, but the connection fails.

Consider the following example to demonstrate this scenario:

```
openssl s_client -connect localhost:<ldaps_portnumber>
connect: Connection refused
connect:errno=146
```

Solution

A possible solution is to check the correct value of LDAPS number in config.ldif file.

26.12.3 Scenario 2- Verify Return Code: 18 (Self Signed Certificate)

When you receive an error code 18, this implies your SSL client program failed to establish the secure connection (https) with the server due to certificate chain verification failure. The server that you are using is a self-signed certificate, and you must use a certificate chain.

Consider the following example to demonstrate this scenario:

```
openssl s_client -connect localhost:<ldaps-port-number>
CONNECTED(00000004)
depth=0 /C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
verify error:num=18:self signed certificate
verify return:1
depth=0 /C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
verify return:1
---
Certificate chain
 0 s:/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
   i:/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDBjCCAsSgAwIBAgIETxRMvTALBgcqhkjOOAQDBQAwZjELMAkGA1UEBhMCY2Ex
EzARBgNVBAgTCkNhbGlmb3JuaWExCzAJBgNVBAcTAlNGMQ8wDQYDVQQKEwZPcmFj
bGUxDTALBgNVBAsTBGxkYXAxFTATBgNVBAMTDHNlcnZlciBhZG1pbjAeFw0xMjAx
```

```
MTYxNjEzNDlaFw0xMjA0MTUxNjEzNDlaMGYxCzAJBgNVBAYTAmNhMRMwEQYDVQQI
EwpDYWxpZm9ybmlhMQswCQYDVQQHEwJTRjEPMA0GA1UEChMGT3JhY2xlMQ0wCwYD
VQQLEwRsZGFwMRUwEwYDVQQDEwxzZXJ2ZXIgYWRtaW4wggG4MIIBLAYHKoZIzjgE
ATCCAR8CgYEA/X9TgR11EilS30qcLuzk5/YRt1I870QAwx4/gLZRJmlFXUAiUftZ
PY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWkn5/oBHsQIsJPu6nX/rfGG/g7
V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFIAccCFQCXYFCP
FSMLzLKSuYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o66oL5V0wL
PQeCZ1FZV4661FlP5nEHEIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8jSjgo64e
K7OmdZFuo38L+iE1YvH7YnoBJDvMpPG+qFGQiaiD3+Fa5Z8GkotmXoB7VSVkAUw7
/s9JKgOBhQACgYEAw+2EIpmwy0rqtHbNb6gxbEtW0hplXXQdHEQp24brde1jt1qv
LDz/c8KR+fVxqvTxAmurGt1qbrhjXcUxi1KdaLnLnLXTCoD+ZLQU+F6B/TNmfrxb
AJmHtmoZsFtNCBTC++FClXtconKyXjEWnKMw7fEb+gNY3eTUrcyIpa/YEbYwCwYH
KoZIzjgEAwUAAy8AMCwCFEtf5+J77Q/5fI6bZ7k3D1rdbw6UAhQkWGmp8VOiMdUg
5K4wK7Y7cC0wSQ==
-----END CERTIFICATE-----
subject=/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
issuer=/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
Acceptable client certificate CA names
/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=user.41
/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
SSL handshake has read 1594 bytes and written 312 bytes
---
New, TLSv1/SSLv3, Cipher is EDH-DSS-DES-CBC3-SHA
Server public key is 1024 bit
SSL-Session:
    Protocol  : TLSv1
    Cipher    : EDH-DSS-DES-CBC3-SHA
    Session-ID: 4F16C3F27655013F71AE2120134A8D1AFE966A1D9233618507DEFE9C607417AA
    Session-ID-ctx:
    Master-Key:
57BDB7FCA9A293E65274AA7CDD0E7CC48AA227806FC2B54C9F9E36BB26D32943FC115CE4FF9A605B6B6BD2370
26F3D0E
    Key-Arg   : None
    Start Time: 1326892018
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
```

Solution

You must import in the server key store, signed certificate reply, and CA certificate.

26.12.4 Scenario 3 - Verify Return Code: 0 (ok)

If a connection is successfully established with an SSL server, then you receive a return code `0`. This implies that any data received from the server is displayed and any key presses will be sent to the server. In addition, the certificate chain in use is also displayed.

Consider the following example to demonstrate a working session:

```
openssl s_client -connect localhost:8636 -verify 250 \
-key $SERVER_SSL/config/keystore   -CApath $CA_SSL -CAfile ca-cert.pem

-key is specifying the path to the server keystore
-CAPath/-CAfile allows to locate CA certificate (pem format)

verify depth is 250
CONNECTED(00000004)
depth=1 /C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
verify return:1
```

```
depth=0 /C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
verify return:1
---
Certificate chain
 0 s:/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
   i:/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
 1 s:/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
   i:/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDYDCCAsmgAwIBAgIFAJbW4rkwDQYJKoZIhvcNAQEFBQAwaTELMAkGA1UEBhMC
RlIxDzANBgNVBAgTBkZyYW5jZTERMA8GA1UEBxMIR3Jlbm9ibGUxDzANBgNVBAoT
Bk9yYWNsZTEMMAoGA1UECxMDT1VEMRcwFQYDVQQDEw5DQSBDZXJ0aWZpY2F0ZTAe
Fw0xMjAxMTcxMDQ5MjdaFw0xMjA0MTcxMDQ5MjdaMGYxCzAJBgNVBAYTAmNhMRMw
EQYDVQQIEwpDYWxpZm9ybmlhMQswCQYDVQQHEwJTRjEPMA0GA1UEChMGT3JhY2xl
MQ0wCwYDVQQLEwRsZGFwMRUwEwYDVQQDEwxzZXJ2ZXIgYWRtaW4wggG3MIIBLAYH
KoZIzjgEATCCAR8CgYEA/X9TgR11EilS30qcLuzk5/YRt1I870QAwx4/gLZRJmlF
XUAiUftZPY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWkn5/oBHsQIsJPu6nX
/rfGG/g7V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFIAccC
FQCXYFCPFSMLzLKSuYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o6
6oL5V0wLPQeCZ1FZV4661FlP5nEHEIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8
jSjgo64eK7OmdZFuo38L+iE1YvH7YnoBJDvMpPG+qFGQiaiD3+Fa5Z8GkotmXoB7
VSVkAUw7/s9JKgOBhAACgYA8N/yzB5rrvNOPhOrea1RNCRePn0bMvXkDpfUs8dpH
z1qQog4soloAhojIYJYA3OGqKr3ryNnfB0B8lePQ1ZaJgkURqOjiVKF6xv5FmnuM
C1uwiTfr/9IKijiy8oCKKKSLTB5lY3Rk0o03D+LrqgLp27A41WvvhGo4djBqXse1
OTANBgkqhkiG9w0BAQUFAAOBgQBzTpgFc1YCpo8QKeoDBRag4tn2y8BzkeLeLMgy
gQAYCGNjJjrV0ChYKMJnqLPCrP9+/Otyj9ZByn9+T1Jx9/khuh9oNXCwF5FUE5VE
gkn3kPo1LdLBqKpfUSeFcYNJDQDhtThVwEq05Ifm+JuCCM4J3BbFuZpJM5xnbcIZ
mjcn5w==
-----END CERTIFICATE-----
subject=/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
issuer=/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
---Verify return code: 0 (ok)
Acceptable client certificate CA names
/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
/C=fr/ST=Isere/L=Montbonnot/O=Oracle/OU=ldap/CN=server_8839
---
SSL handshake has read 2179 bytes and written 312 bytes
---
New, TLSv1/SSLv3, Cipher is EDH-DSS-DES-CBC3-SHA
Server public key is 1024 bit
SSL-Session:
    Protocol  : TLSv1
    Cipher    : EDH-DSS-DES-CBC3-SHA
    Session-ID: 4F16C59B172D329E44AF199B4E49B14E54163AAF783A68FBD48556FCB06A9238
    Session-ID-ctx:
    Master-Key:
21CC1BBF638FFDAF16E5DBAB337728D029F0125D483636EF7590BE3005DDA96AEAF60DE88172DE925806F633E
B09ACBE
    Key-Arg   : None
    Start Time: 1326892443
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
```

26.12.5 Scenario 4 - SSLHandshakeException

When you try to establish a server secure connection, the ldapsearch issues the error message.

The error message is as follows:

```
ldapsearch  -p 7636 -D "cn=Directory Manager" -w secret12  -P config/keystore.p12
-Z -b dc=example,dc=com uid=user.0 Cannot send the simple bind request:
SSLHandshakeException(sun.security.validator.ValidatorException: PKIX path
 building failed: sun.security.provider.certpath.SunCertPathBuilderException:
 unable to find valid certification path to requested target)
```

This error appears because the server certificate is self signed certificate and not a certificate chain. You will receive an error code `18`.

The following demonstrates an example of this process.

```
openssl s_client -connect localhost:7636
CONNECTED(00000004)
depth=0 /C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
verify error:num=18:self signed certificate
verify return:1
depth=0 /C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
verify return:1
---
Certificate chain
 0 s:/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
   i:/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDBjCCAsSgAwIBAgIETxRMvTALBgcqhkjOOAQDBQAwZjELMAkGA1UEBhMCY2Ex
EzARBgNVBAgTCkNhbGlmb3JuaWExCzAJBgNVBAcTAlNGMQ8wDQYDVQQKEwZPcmFj
bGUxDTALBgNVBAsTBGxkYXAxFTATBgNVBAMTDHNlcnZlciBhZG1pbjAeFw0xMjAx
MTYxNjEzNDlaFw0xMjA0MTUxNjEzNDlaMGYxCzAJBgNVBAYTAmNhMRMwEQYDVQQI
EwpDYWxpZm9ybmlhMQswCQYDVQQHEwJTRjEPMA0GA1UEChMGT3JhY2xlMQ0wCwYD
VQQLEwRsZGFwMRUwEwYDVQQDEwxzZXJ2ZXIgYWRtaW4wggG4MIIBLAYHKoZIzjgE
ATCCAR8CgYEA/X9TgR11EilS30qcLuzk5/YRt1I870QAwx4/gLZRJmlFXUAiUftZ
PY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWkn5/oBHsQIsJPu6nX/rfGG/g7
V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFIAccCFQCXYFCP
FSMLzLKSuYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o66oL5V0wL
PQeCZ1FZV4661FlP5nEHEIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8jSjgo64e
K7OmdZFuo38L+iE1YvH7YnoBJDvMpPG+qFGQiaiD3+Fa5Z8GkotmXoB7VSVkAUw7
/s9JKgOBhQACgYEAw+2EIpmwy0rqtHbNb6gxbEtW0hplXXQdHEQp24brde1jt1qv
LDz/c8KR+fVxqvTxAmurGt1qbrhjXcUxi1KdaLnLnLXTCoD+ZLQU+F6B/TNmfrxb
AJmHtmoZsFtNCBTC++FClXtconKyXjEWnKMw7fEb+gNY3eTUrcyIpa/YEbYwCwYH
KoZIzjgEAwUAAy8AMCwCFEtf5+J77Q/5fI6bZ7k3D1rdbw6UAhQkWGmp8VOiMdUg
5K4wK7Y7cC0wSQ==
-----END CERTIFICATE-----
subject=/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
issuer=/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
Acceptable client certificate CA names
/C=FR/ST=France/L=Grenoble/O=Oracle/OU=OUD/CN=CA Certificate
/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=user.41
/C=ca/ST=California/L=SF/O=Oracle/OU=ldap/CN=server admin
---
SSL handshake has read 1594 bytes and written 312 bytes
---
New, TLSv1/SSLv3, Cipher is EDH-DSS-DES-CBC3-SHA
Server public key is 1024 bit
SSL-Session:
    Protocol  : TLSv1
    Cipher    : EDH-DSS-DES-CBC3-SHA
    Session-ID: 4F16C3F27655013F71AE2120134A8D1AFE966A1D9233618507DEFE9C607417AA
    Session-ID-ctx:
    Master-Key:
57BDB7FCA9A293E65274AA7CDD0E7CC48AA227806FC2B54C9F9E36BB26D32943FC115CE4FF9A605B6B6BD2370
```

```
26F3D0E
    Key-Arg   : None
    Start Time: 1326892018
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
```

Solution

To fix this issue:

1. Import the CA certificate into the server keystore.

```
keytool -importcert -alias ca-cert -keystore config/keystore.p12 -storetype PKCS12 -
file $CA_SSL/ca-cert.pem
Enter keystore password:
Owner: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Issuer: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Serial number: 96b69e65
Valid from: Wed Jan 04 15:51:37 MET 2012 until: Mon Sep 04 16:51:37 MEST 2428
Certificate fingerprints:
         MD5:  D0:5B:C8:2A:3D:3B:09:07:5A:29:62:E3:27:99:4E:D4
         SHA1: E4:C9:BB:B7:5B:49:C7:7E:BF:8B:C3:C3:DC:DF:29:E7:74:A0:66:03
         Signature algorithm name: SHA1withRSA
         Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

2. Import the signed server certificate reply into the server keystore.

```
keytool -importcert -trustcacerts -alias server-cert
-keystore config/keystore.p12 -storetype PKCS12
-file server-cert.pem Enter keystore password:
Certificate reply was installed in keystore
```

3. List certificates in the LDAP server keystore.

```
keytool -list -keystore config/keystore.p12 -storepass secret12 -v

Keystore type: PKCS12
Keystore provider: SUN

Your keystore contains 2 entries

Alias name: ca-cert
Creation date: Jan 18, 2012
Entry type: trustedCertEntry

Owner: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Issuer: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Serial number: 96b69e65
Valid from: Wed Jan 04 15:51:37 MET 2012 until: Mon Sep 04 16:51:37 MEST 2428
Certificate fingerprints:
         MD5:  D0:5B:C8:2A:3D:3B:09:07:5A:29:62:E3:27:99:4E:D4
         SHA1: E4:C9:BB:B7:5B:49:C7:7E:BF:8B:C3:C3:DC:DF:29:E7:74:A0:66:03
         Signature algorithm name: SHA1withRSA
         Version: 3
```

4. Verify the connection with a `ldapsearch` request over SSL.

```
ldapsearch  -p 7636 -D "cn=Directory Manager" -w secret12  -P config/keystore.p12 -Z
-b dc=example,dc=com uid=user.0
dn: uid=user.0,ou=People,dc=example,dc=com
postalAddress: Aaccf Amar$01251 Chestnut Street$Panama City, DE  50369
postalCode: 50369
uid: user.0
```

```
description: This is the description for Aaccf Amar.
userPassword: {SSHA}vVIy4fjEUyt0L8GSVzX+VrJKEgGASLkeCvL1ng==
employeeNumber: 0
initials: ASA
givenName: Aaccf
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: top
pager: +1 779 041 6341
mobile: +1 010 154 3228
cn: Aaccf Amar
telephoneNumber: +1 685 622 6202
sn: Amar
street: 01251 Chestnut Street
homePhone: +1 225 216 5900
mail: user.0@maildomain.net
l: Panama City
st: DE
```

5. Access the log.

```
[18/Jan/2012:16:39:24 +0100] CONNECT conn=1 from=127.0.0.1:46726 to=127.0.0.1:7636
protocol=LDAPS
[18/Jan/2012:16:39:24 +0100] BIND REQ conn=1 op=0 msgID=1 type=SIMPLE
dn="cn=Directory Manager"
[18/Jan/2012:16:39:24 +0100] BIND RES conn=1 op=0 msgID=1 result=0
authDN="cn=Directory Manager,cn=Root DNs,cn=config" etime=31
[18/Jan/2012:16:39:24 +0100] SEARCH REQ conn=1 op=1 msgID=2 base="dc=example,dc=com"
scope=wholeSubtree filter="(uid=user.0)" attrs="ALL"
[18/Jan/2012:16:39:24 +0100] SEARCH RES conn=1 op=1 msgID=2 result=0 nentries=1
etime=18
[18/Jan/2012:16:39:24 +0100] UNBIND REQ conn=1 op=2 msgID=3
[18/Jan/2012:16:39:24 +0100] DISCONNECT conn=1 reason="Client Disconnect"
```

26.12.6 Scenario 5 - SASL EXTERNAL Bind Request Could Not Be Processed

When performing OUD SASL client external authentication over SSL, you get error message.

The following error message appears:

```
ldapsearch -p 7636 -Z -K /export/home/oud/security/client/config/keystore
-W secret12 -P /export/home/oud/security/client/config/truststore
 --trustStorePassword secret12 -N user.41-cert --useSASLExternal
-b dc=example,dc=com uid=user.0
The SASL EXTERNAL bind attempt failed
Result Code:  49 (Invalid Credentials)
```

When you view the access log, then the following message is shown:

```
CONNECT conn=2 from=127.0.0.1:46763 to=127.0.0.1:7636 protocol=LDAPS
[18/Jan/2012:17:48:44 +0100] BIND REQ conn=2 op=0 msgID=1 type=SASL
 mechanism=EXTERNAL dn=""
[18/Jan/2012:17:48:44 +0100] BIND RES conn=2 op=0 msgID=1 result=49
authFailureID=1245310 authFailureReason="The SASL EXTERNAL bind request could not
be processed because the client did not present a certificate chain during
SSL/TLS negotiation" etime=6
[18/Jan/2012:17:48:44 +0100] DISCONNECT conn=2 reason="Client Disconnect"
```

This error appears because the client certificate is not a valid certificate chain.

Solution

To fix this issue:

1. Import the CA certificate into the client keystore.

```
keytool -importcert -alias ca-cert -keystore config/keystore \
-storetype JKS -file $CA_SSL/ca-cert.pem
Enter keystore password:
Owner: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Issuer: CN=CA Certificate, OU=OUD, O=Oracle, L=Grenoble, ST=France, C=FR
Serial number: 96b69e65
Valid from: Wed Jan 04 15:51:37 MET 2012 until: Mon Sep 04 16:51:37 MEST 2428
Certificate fingerprints:
         MD5:  D0:5B:C8:2A:3D:3B:09:07:5A:29:62:E3:27:99:4E:D4
         SHA1: E4:C9:BB:B7:5B:49:C7:7E:BF:8B:C3:C3:DC:DF:29:E7:74:A0:66:03
         Signature algorithm name: SHA1withRSA
         Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

2. Import the user signed reply certificate into the client keystore.

```
keytool -importcert -trustcacerts -alias user.41-cert
-keystore config/keystore -storetype JKS -file user.41-cert.pem
-storepass secret12
Certificate reply was installed in keystore
```

3. Run the `ldap` command.

```
ldapsearch -p 7636 -Z -K /export/home/oud/security/client/config/keystore
-W secret12 -P /export/home/oud/security/client/config/truststore
--trustStorePassword secret12 -N user.41-cert --useSASLExternal
 -b dc=example,dc=com uid=user.0
dn: uid=user.0,ou=People,dc=example,dc=com
postalAddress: Aaccf Amar$01251 Chestnut Street$Panama City, DE  50369
postalCode: 50369
uid: user.0
description: This is the description for Aaccf Amar.
employeeNumber: 0
initials: ASA
givenName: Aaccf
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: top
pager: +1 779 041 6341
mobile: +1 010 154 3228
cn: Aaccf Amar
telephoneNumber: +1 685 622 6202
sn: Amar
street: 01251 Chestnut Street
homePhone: +1 225 216 5900
mail: user.0@maildomain.net
l: Panama City
st: DE
```

4. Validate the log.

```
[18/Jan/2012:18:04:49 +0100] CONNECT conn=3 from=127.0.0.1:46777
 to=127.0.0.1:7636 protocol=LDAPS
[18/Jan/2012:18:04:49 +0100] BIND REQ conn=3 op=0 msgID=1
type=SASL mechanism=EXTERNAL dn=""
[18/Jan/2012:18:04:49 +0100] BIND RES conn=3 op=0 msgID=1
result=0 authDN="uid=user.41,ou=People,dc=example,dc=com" etime=37
```

```
[18/Jan/2012:18:04:49 +0100] SEARCH REQ conn=3 op=1 msgID=2
 base="dc=example,dc=com" scope=wholeSubtree filter="(uid=user.0)" attrs="ALL"
[18/Jan/2012:18:04:49 +0100] SEARCH RES conn=3 op=1 msgID=2 result=0
nentries=1 etime=15
[18/Jan/2012:18:04:49 +0100] UNBIND REQ conn=3 op=2 msgID=3
[18/Jan/2012:18:04:49 +0100] DISCONNECT conn=3 reason="Client Disconnect"
```

26.13 Debugging SSL or TLS Using Java Debug Information

You can troubleshoot network Traffic for SSL or TLS connections using Java debug information.

There are situations when the only way to analyze SSL is to trace network access. Oracle Unified Directory allows you to debug SSL by adding `-Djavax.net.debug=all` option to the server in the *config/java.properties* file.

A sample debug output is as follows:

```
server.core.DirectoryServer (alert type org.opends.server.DirectoryServerStarted,
 alert ID 458887):  The Directory Server has started successfully
***
found key for : server-cert
chain [0] = [
[
  Version: V3
  Subject: CN=server admin, OU=ldap, O=mycompany, L=City1, ST=Country1, C=ca
  Signature Algorithm: SHA1withRSA, OID = 1.2.840.113549.1.1.5

  Key:  SunPKCS11-Solaris DSA public key, 1024 bits (id 22714576, session object)
  y: 13758517829882967277399226271740078303525267606775373025890213388747276573
75961628650688647577510816321283250872882407370491996059918680923417848100018 23
89355776410228200735673010501146203945919143729329772551286385346818351986257 75
05401958362086546885405080570540575677103845462467633475547155894544465662390
  p: 17801190547854226652823756245015999014523215636912067427327445031444286578873
70207706126952521234630795671567847784664499706507709207278570500096683881440
34129745221171818506047231150039301079959358067395348717066319802262019714966 5
24135060945913707594956514672855690606794135837542707371727429551343320695239
  q: 864205495604807476120572616017955259175325408501
  g: 17406820753240209518581198012352343653860449079456135097849583104059995
34884558231478515974089409507253077970949157594923683005742524387610370844 7
34671801488761181030830437549851909834726015504946913294880833954923138500 00
36164648264460849230407872181895999905649609776936801774927370896200668918 79
56744210730
  Validity: [From: Mon Jan 16 17:15:45 MET 2012,
             To: Mon Apr 16 18:15:45 MEST 2012]
  Issuer: CN=CA Certificate, OU=OUD, O=mycompany, L=City2, ST=Country2, C=FR
  SerialNumber: [96d4f0dc]

]
  Algorithm: [SHA1withRSA]
  Signature:
0000: 72 F6 7E 93 2B 87 B9 C7   39 51 4C D2 A7 B0 AA 36  r...+...9QL....6
0010: B8 0F BA C4 6E 43 70 72   81 50 09 7A 88 05 16 A2  ....nCpr.P.z....
0020: 1C 96 C2 49 B3 0A F9 AB   2B 4B 8D 59 4C BA 58 C9  ...I....+K.YL.X.
0030: EF B9 48 58 A7 C5 BB B5   0E 64 51 CF BC 58 DA 71  ..HX.....dQ..X.q
0040: E1 F7 2E A4 1D 1B FC D5   4F B2 70 B0 78 F4 FB E6  ........O.p.x...
0050: C4 6A 6A E0 DE B0 F5 98   7B 09 A9 A4 9D 17 4C F5  .jj...........L.
0060: 9F 06 07 E1 09 81 77 9E   41 3C 02 4C FB D8 94 ED  ......w.A<.L....
0070: 36 6A 65 5A 96 2C AE A4   86 83 66 63 BC 3C 8C 47  6jeZ.,....fc.<.G

]
```

The preceding information is provided in addition to the Oracle Unified Directory debug log text.

The following topics describe how to work with SSL debug recording:

- Enabling SSL Debug Recording
- Disabling SSL Debug Recording

26.13.1 Enabling SSL Debug Recording

You can enable SSL debug recording by updating `start-ds.java-args` property.

Perform the following steps to enable SSL debug recording:

1. Update the `start-ds.java-args` property in the *config/java.properties* file with:

   ```
   start-ds.java-args=-server -Djavax.net.debug=all
   ```

2. Run the `dsjavaproperties` command as described in dsjavaproperties
3. Stop the server instance using the `stop-ds` command.
4. Restart the server instance using the `start-ds` command.

> **Note:**
>
> The SSL debug information is logged in the *logs/server.out* file.

26.13.2 Disabling SSL Debug Recording

You can disable SSL debug recording by deleting `-Djavax.net.debug=all` property.

Perform the following steps to disable SSL debug recording:

1. Delete the `-Djavax.net.debug=all` property from *java.properties* file.

   ```
   start-ds.java-args=-server
   ```

2. Run the `dsjavaproperties` command as described in dsjavaproperties
3. Stop the server instance using the `stop-ds` command.
4. Restart the server instance using the `start-ds` command.

26.14 Controlling Connection Access Using Allowed and Denied Rules

You can control connection handlers that are responsible for accepting connections to the server.

These topics provide description on the types of connection handlers and their properties and syntax:

- About Connection Handlers
- Property Syntax of Allowed and Denied Client Rules

- Configuring Allowed and Denied Client Rules

26.14.1 About Connection Handlers

You can use connection handler allowed and denied client rules to control which hosts can make TCP connections to the server. Connection handlers are responsible for accepting connections to the server.

The different types of connection handlers and their configuration properties are presented in this section and include the following:

- `allowed-client`. Specifies a set of host names or address masks that determine the clients that are allowed to establish connections to this Connection Handler. Valid values include a host name, a fully qualified domain name, a domain name, an IP address, or a subnetwork with subnetwork mask.
- `denied-client`. Specifies a set of host names or address masks that determine the clients that are not allowed to establish connections to this Connection Handler. Valid values include a host name, a fully qualified domain name, a domain name, an IP address, or a subnetwork with subnetwork mask. If both allowed and denied client masks are defined and a client connection matches one or more masks in both lists, then the connection is denied. If only a denied list is specified, then any client not matching a mask in that list is allowed.

> **Note:**
>
> Both IPv4 and IPv6 addresses are supported.

26.14.2 Property Syntax of Allowed and Denied Client Rules

The `allowed-client` and `denied-client` properties share the same syntax to perform pattern matching against IP (IPv4 or IPv6) addresses and host names.

The following syntaxes are supported:

- IP address - The IP address of the clients to be allowed or denied can be specified in the rule. For example:

```
ds-cfg-denied-client:   192.168.5.6
ds-cfg-allowed-client:  2001:fecd:ba23:cd1f:dcb1:1010:9234:4088
```

- IP address with CIDR notation - A range of IP addresses can be allowed or denied by specifying an IP address using CIDR notation. For example:

```
ds-cfg-denied-client:   192.168.5.6/28
ds-cfg-allowed-client:  2001:0db8:1234::/48
```

  The first denies clients in the range `192.168.5.0 - 192.168.5.15` and the second allows clients in the range `2001:0db8:1234:0000:0000:0000:0000:0000 - 2001:0db8:1234:ffff:ffff:ffff:ffff:ffff`.

- IP address with '*' notation - A range of IP addresses (IPv4 only) can be allowed or denied by specifying an IP address with a '*' character to match parts of the IP address. For example:

```
ds-cfg-denied-client:   192.168.5.*
ds-cfg-allowed-client:   129.45.*.*
```

The first example denies clients with IP addresses starting with 192.168.5 and the second allows clients with IP address starting with 129.45. Notice that the second example uses multiple match characters. To allow all IP addresses to match, the rule would look like:

```
ds-cfg-denied-client:   *.*.*.*
```

- DNS names - Clients can be restricted by DNS name. For example to restrict clients with the host name `foo.example.com`, enter:

  ```
  ds-cfg-denied-client:   foo.example.com
  ```

- DNS names with pattern matching - This is similar to IP address pattern matching. The property can specify the '*' character to match parts of the DN name:

  ```
  ds-cfg-allowed-client: foo.*.test.com
  ```

  The property allows clients with DN names such as: `foo.bar.test.com` or `foo.foobar.test.com`. To only match DNS names ending in a suffix the property would be:

  ```
  ds-cfg-allowed-client: .example.com
  ```

  This property allows clients with DNS names such as: `test.example.com` or `test.me.example.com`.

> **Note:**
>
> Be careful when you use the DNS properties because the host name resolution depends on the server name service configuration.

26.14.3 Configuring Allowed and Denied Client Rules

Each connection handler needs to have its own set of rules. You can use `dsconfig` command to manage the allowed and denied properties for each connection handler

Below example lists the set of rules:

```
dn: cn=LDAP Connection Handler,cn=Connection Handlers,cn=config
objectClass: top
objectClass: ds-cfg-connection-handler
objectClass: ds-cfg-ldap-connection-handler
cn: LDAP Connection Handler
ds-cfg-java-class: org.opends.server.protocols.ldap.LDAPConnectionHandler
ds-cfg-enabled: true
ds-cfg-listen-address: 0.0.0.0
ds-cfg-listen-port: 389
ds-cfg-accept-backlog: 128
ds-cfg-allow-ldap-v2: true
ds-cfg-keep-stats: true
ds-cfg-use-tcp-keep-alive: true
ds-cfg-use-tcp-no-delay: true
ds-cfg-allow-tcp-reuse-address: true
ds-cfg-send-rejection-notice: true
ds-cfg-max-request-size: 5 megabytes
ds-cfg-max-blocked-write-time-limit: 2 minutes
ds-cfg-num-request-handlers: 2
ds-cfg-allow-start-tls: false
ds-cfg-use-ssl: false
ds-cfg-ssl-client-auth-policy: optional
ds-cfg-ssl-cert-nickname: server-cert
```

```
ds-cfg-denied-client: *.example.com
ds-cfg-denied-client:  129.45.*.*
ds-cfg-denied-client:   192.168.5.6

dn: cn=LDAPS Connection Handler,cn=Connection Handlers,cn=config
objectClass: top
objectClass: ds-cfg-connection-handler
objectClass: ds-cfg-ldap-connection-handler
cn: LDAPS Connection Handler
ds-cfg-java-class: org.opends.server.protocols.ldap.LDAPConnectionHandler
ds-cfg-enabled: true
ds-cfg-listen-address: 0.0.0.0
ds-cfg-listen-port: 636
ds-cfg-accept-backlog: 128
ds-cfg-allow-ldap-v2: true
ds-cfg-keep-stats: true
ds-cfg-use-tcp-keep-alive: true
ds-cfg-use-tcp-no-delay: true
ds-cfg-allow-tcp-reuse-address: true
ds-cfg-send-rejection-notice: true
ds-cfg-max-request-size: 5 megabytes
ds-cfg-max-blocked-write-time-limit: 2 minutes
ds-cfg-num-request-handlers: 2
ds-cfg-allow-start-tls: false
ds-cfg-use-ssl: true
ds-cfg-ssl-client-auth-policy: optional
ds-cfg-ssl-cert-nickname: server-cert
ds-cfg-key-manager-provider: cn=JKS,cn=Key Manager Providers,cn=config
ds-cfg-trust-manager-provider: cn=JKS,cn=Trust Manager Providers,cn=config
ds-cfg-allowed-client: .example.com
ds-cfg-allowed-client: foo.*.test.com
ds-cfg-allowed-client: 192.168.6.7/22
```

Use the `dsconfig` command to manage the allowed and denied properties for each connection handler. For example:

```
$ dsconfig -n -X -p 4444 -D "cn=directory manager" -j pwd-file \
  set-connection-handler-prop   --handler-name "LDAPS Connection Handler" \
  --set denied-client:.example.com \
  --set allowed-client:192.168.1.6/17
```

> **Note:**
>
> Denied rules are applied before the allowed rules.

26.15 Configuring Unlimited Strength Cryptography

You must download the Java Cryptography Extension Unlimited Strength Jurisdiction policy files for missing cryptography support to configure unlimited strength cryptography.

To download and install the policy file for configuring unlimited strength cryptography:

1. Download the Java Cryptography Extension Unlimited Strength Jurisdiction policy files from the following Web page

   http://www.oracle.com/technetwork/java/javase/downloads/index.html

2. Perform the installation instructions described in the README.txt file that is part of the downloaded zip.

   Java Cryptography Extension Unlimited Strength Jurisdiction policy files are now installed.

3. Stop the Oracle Unified Directory server, and then restart.

26.16 Configuring TLS Protocols and Cipher Suites for OUDSM to OUD Communication

If Oracle Unified Directory server is not using system default protocols and cipher suites, then you must configure OUDSM to use the protocols and cipher suites that Oracle Unified Directory server supports.

See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to learn about system default values that are used for secured communication.

To configure TLS protocols and cipher suites for OUDSM to OUD server communication, perform these steps:

1. To configure protocol version, set OUDSM weblogic system property as:
   `weblogic.security.SSL.minimumProtocolVersion=[protocol]`

   Consider the following example to set `protocol` version to `TLSv1.1`:

   ```
   weblogic.security.SSL.minimumProtocolVersion=TLSv1.1
   ```

   See Using the weblogic.security.SSL.minimumProtocolVersion System Property in *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* to know about protocols used for SSL connection.

   > **Note:**
   >
   > You need to restart OUDSM weblogic server after you set the protocol version.

2. To configure cipher suites, use WLST command `setCipherSuites()` in weblogic server's SSL configuration.

   See Setting Cipher Suites Using WLST: An Example in *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* to know about setting cipher suites using WLST script.

27

Configuring Security Between the Proxy and the Data Source

You can configure security between the proxy and the remote LDAP servers.
The following topics describe how to configure security between the proxy and the remote LDAP servers:

- About Security Between the Proxy and Remote LDAP Servers
- About Proxy Manages Secure Connections
- Understanding the Modes of Secure Connection
- Configuring Security Between Proxy and Data Source Using `dsconfig`

You can configure security between the proxy and the remote LDAP servers as follows:

- During installation of the proxy by using the `oud-proxy-setup` GUI. For more information, see "Setting Up the Proxy Server by Using the GUI" in *Installing Oracle Unified Directory*

  .
- After the proxy installation, by using the `dsconfig` command in interactive mode. For general information about using the `dsconfig` command, see Managing the Server Configuration Using `dsconfig`.

27.1 About Security Between the Proxy and Remote LDAP Servers

For security management, network groups can be enabled to classify incoming client connections. You can use network groups to restrict operations that can be performed, based on how the connection has been classified.

Use this functionality, for example, to restrict access to clients that connect from a specified IP address only. For more information, see Configuring Network Groups Using `dsconfig`.

For secure client authentication between the proxy and remote LDAP servers, the certificate of the proxy must be imported into the truststore of each remote LDAP server. In this case, you must configure a keystore manually. For details, see Configuring Key Manager Providers.

The proxy security does not bypass the back-end ACI.

27.2 About Proxy Manages Secure Connections

The proxy manages the security with the client and with the directory server, and supports both SSL and StartTLS.

When you configure security, you must specify how the proxy connects to the remote LDAP server by indicating if the proxy should use SSL `always`, `never`, or `user`. If you specify `always`, the connection with the remote LDAP server will always be secured using SSL, regardless of how the client connects to the proxy. If you specify `never`, the connection between the proxy

and the remote LDAP directory server will not be secured, regardless of whether the client connects to the proxy with a secure connection. If specify `user`, the security between the proxy and the remote LDAP directory servers will be the same as the security between the client and the proxy. For example, if the client connects over SSL, the connection with the remote LDAP server will also use SSL. One notable exception is if the client connects using StartTLS, in which case the proxy will connect to the remote LDAP servers using SSL.

> **Note:**
>
> If you want the modifications of the privileges of a user used by proxy to bind on the remote server to take effect, then you must set the `maintain-authenticated-users` flag to `true` on the remote server. By default, it is set to `false`.
>
> Be aware that for an open connection, which is bound with a determined `authDN`, importing that entry with `dn: authDN` using `import-ldif` command does not modify the properties (access rights, privileges, and so on) of that `authDN` in those already established connections. The new properties for the `authDN` as a result of `import-ldif` are effective only for new binds as `authDN`. In this scenario, setting `maintain-authenticated-users:true` does not help. Consider the following example.
>
> For example, in a proxy scenario if the bind mode for the remote LDAP server is set as `use-specific-id` and the `remote-ldap-server-bind-dn` is `cn=my_proxy_manager,dc=com`, then the proxy keeps a pool of open connections with the remote LDAP server bound as `authDN='cn=my_proxy_manager,dc=com'`. Now, if the user entry `cn=my_proxy_manager,dc=com` stored in the remote LDAP server does not have `password-reset` privilege, then the operation to modify the password that arrives at proxy server fails, because of insufficient privileges. So, you might attempt to re-load the data in the remote LDAP server by importing an ldif file that would add the required `password-reset` privilege to the `cn=my_proxy_manager,dc=com` user entry. However, the new privilege will still not be taken into account for those connections already opened.

For more information see Understanding the Modes of Secure Connection.

27.3 Understanding the Modes of Secure Connection

Secure connection with remote LDAP servers is handled by proxy.

The proxy handles connections to the remote LDAP servers in three SSL security modes:

- `always`
- `never`
- `user`

You can view or edit these settings using the `dsconfig --advanced` command. Choose Extension from the main menu.

The `remote-ldap-server-ssl-policy` property manages the three SSL security modes.

When the `remote-ldap-server-ssl-policy` property is set to `always` or user, the proxy needs to trust the remote LDAP servers. To achieve this, you must manually import the certificates of each remote LDAP server into the proxy's truststore.

The following topics explain the modes of secure connection:

- About `always` Secure Mode
- About `never` Secure Mode
- About `user` Secure Mode

27.3.1 About `always` Secure Mode

With the `remote-ldap-server-ssl-policy` property set to `always`, all connections made from the proxy to the remote LDAP servers are fully secure SSL connections, regardless how the client connects to the proxy.

In this mode, the pool size refers to one type of connection pool: secure LDAPS connections.

In the `always` secure mode, the certificate of each remote LDAP server must be imported into the proxy's truststore. If there is a large number of back-end LDAP servers that are not Oracle Unified Directory servers, and if certificates were not managed during installation, importing certificates into the truststore of the proxy can be a constraint. For test environment purposes, you can speed up this process by using the `ssl-trust-all` parameter. This parameter requests the proxy to trust all remote LDAP servers.

27.3.2 About `never` Secure Mode

With the `remote-ldap-server-ssl-policy` property set to `never`, none of the connections from the proxy to the remote LDAP servers are secure SSL connections.

In this mode, the monitoring connection by the proxy of the remote LDAP servers is never secure.

In this mode, the pool size refers to one type of connection pool: unsecure LDAP connections.

27.3.3 About `user` Secure Mode

With the `remote-ldap-server-ssl-policy` property set to `user`, incoming requests from clients to the proxy dictate whether the connection between the proxy and remote LDAP servers should be secure, regardless of how the client connects to the proxy.

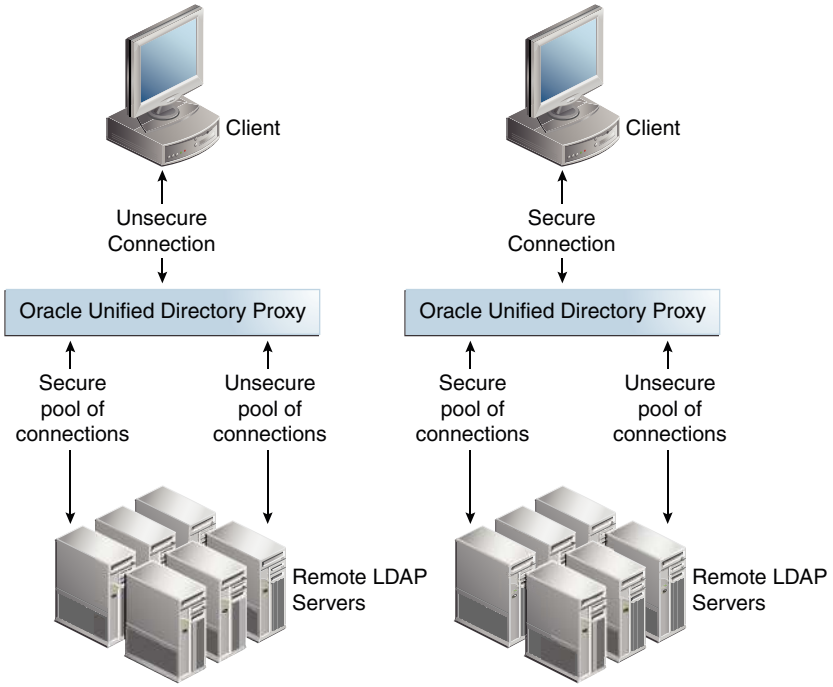
If the incoming client request is secure, whether SSL or StartTLS, the connection from the proxy to the remote LDAP servers is a secure SSL connection.

If the incoming client request is not secure, the connection from the proxy to the remote LDAP servers is not a secure SSL connection.

In this mode, the monitoring connection between the proxy and the remote LDAP servers is never secure.

Two pools of connections are created, one secure and one unsecure. This is shown in Figure 27-1. In the scenario on the left, the client connects to the proxy using an unsecure connection, and the unsecure pool of connections from the proxy to the remote LDAP servers is used. In the scenario on the right, the client connects to the proxy using a secure connection, whether SSL or StartTLS, and the secure SSL pool of connections from the proxy to the remote LDAP servers is used.

Figure 27-1 Connections in the `user` Secure Mode

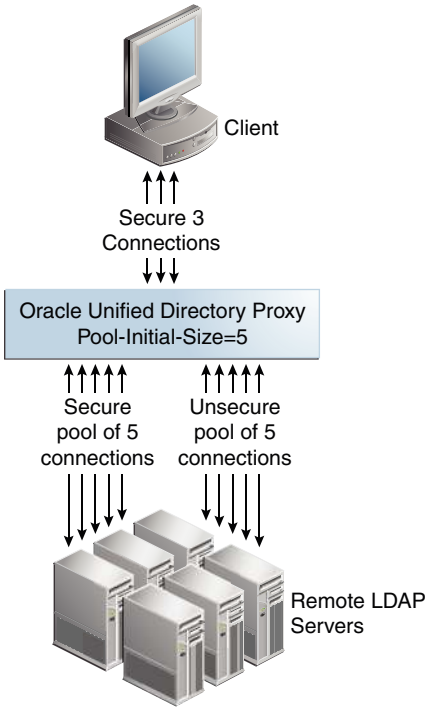


In the `user` mode, the certificate of each remote LDAP server must be imported into the proxy's truststore. If there is a large number of remote LDAP servers that are not Oracle Unified Directory servers, and if certificates were not managed during installation, importing certificates into the truststore of the proxy can be a constraint. In a test environment, you can speed up this process by using the `ssl-trust-all` parameter. This parameter requests the proxy to trust all remote LDAP servers.

When the `remote-ldap-server-ssl-policy` property is set to `user`, the pool size refers to two types of connection pools: unsecure LDAP connections and secure LDAPS connections. If for example the `pool-initial-size` is set to 5 connections, as shown in Figure 27-2, then when the LDAP Extension is initialized, there will be one pool of 5 LDAP connections and one pool of 5 LDAPS connections, or a total of 10 connections. Each pool evolves separately after this initialization, based on parameters set for that pool.

> **Note:**
>
> By default, `pool-initial-size` is set to 10 connections.

Figure 27-2 Multiple Pools of Connections



27.4 Configuring Security Between Proxy and Data Source Using `dsconfig`

The `dsconfig` tool accesses the server over a secured connection with certificate authentication. If you run `dsconfig` in non-interactive mode, as `dsconfig -n`, specification of the trust store parameters depends on whether you run the command locally or remotely.

For more information on running the command locally or remotely, see Using the `dsconfig` Command.

Other configurations using dsconfig are explained in the following sections:

- Configuring Security Between the Proxy and Directory Servers Using `dsconfig`
- About the Configurable LDAP Extension Properties Relevant to Security

27.4.1 Configuring Security Between the Proxy and Directory Servers Using `dsconfig`

This task highlights the main steps required to configure security for connections to remote LDAP servers. Where the process is similar to that provided for configuring security between the proxy and the client, pointers are given to the related procedure.

Perform the following steps to configure security between proxy and directory server using `dsconfig`:

1. If the remote LDAP servers do not require client authentication to be passed from the proxy, proceed directly to step 2.

If the remote LDAP servers require client authentication to be passed from the proxy, perform the following sub-steps:

a. Configure a keystore for remote LDAP server connections.

   To do this, use the Java `keytool` command to generate a certificate on the proxy server. The keystore must be configured manually. For details, see Configuring Key Manager Providers.

   Self-sign the certificate or have the certificate signed by an external certificate authority. For details, see Configuring Key Manager Providers.

b. Configure a key manager provider on the proxy for the keystore for remote LDAP server connections.

   For details, see Configuring Key Manager Providers. This key manager provider can be separate to that used for handling secure connections to clients.

c. If the remote LDAP servers require client authentication, the certificate of the proxy must be imported into the truststore of each remote LDAP server.

   For information about importing and exporting certificates on Oracle Unified Directory, see Configuring Key Manager Providers.

2. For the proxy to establish secure connections with the remote LDAP servers, configure a truststore.

   All remote LDAP servers requiring a secure connection need to have their certificates imported into the proxy truststore. All of these remote LDAP server certificates can be imported into a single proxy truststore or distributed among multiple proxy truststores. You can have as many proxy truststores as there are remote LDAP server certificates to be imported.

   An LDAP proxy extension targeting a secured connection to a remote LDAP data source must reference the appropriate truststore manager in its configuration. This reference enables the LDAP proxy extension to access the imported remote LDAP server certificate, to accept the secure connection.

3. Each truststore requires a proxy trust manager provider.

   To list the proxy trust manager providers, use the `dsconfig list-trust-manager-providers` command. For example:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
   list-trust-manager-providers
   ```

   To create a proxy trust manager provider, use the `dsconfig create-trust-manager-provider` command. For example:

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=Directory Manager"  -j pwd-file -X -n \
   create-trust-manager-provider \
   --provider-name Backend\ Servers \
   --type file-based --set enabled:true \
   --set trust-store-file:/localhost/config/backend-servers-truststore \
   --set trust-store-type:JKS \
   --set trust-store-pin-file:/installPath/config/backend-servers-truststore.pin
   ```

4. Import the certificates of the remote LDAP servers into the proxy truststore.

27.4.2 About the Configurable LDAP Extension Properties Relevant to Security

When managing connections to remote LDAP servers using `dsconfig`, several configurable LDAP Extension security connection properties are available.

For information about managing LDAP extensions, see Configuring Communication With Remote LDAP Servers. Configurable properties that either directly or indirectly relate to security considerations include the following:

- **remote-ldap-server-ssl-policy**

  This important value governs the overall security mode of the connections between the proxy and remote LDAP servers. Its use is covered in the section Understanding the Modes of Secure Connection.

- **pool-increment**

  If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the incremental change of size of each pool is set to `pool-increment`. For more information on this property, see Modifying the Properties of an LDAP Server Extension.

- **pool-initial-size**

  If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the initial size, and minimum size, of each pool is set to `pool-initial-size`. In this case, therefore, there will initially be twice the total number of connections indicated in `pool-initial-size`. For details, see Modifying the Properties of an LDAP Server Extension.

- **pool-max-size**

  If the `remote-ldap-server-ssl-policy` property is set to `user`, two pools of connections are created and the maximum size of each pool is set to `pool-max-size`.

  The default value is 1000 connections. For more information on this property, see Modifying the Properties of an LDAP Server Extension.

- **remote-ldap-server-ssl-port**

  The port number for SSL connections from the proxy to the remote LDAP server.

- **ssl-client-alias**

  When a keystore is created for client authentication, several keys can be stored in it. Use this property to specify which key to use. For more information about keystores, see Getting SSL Up and Running Quickly and Configuring Key Manager Providers.

- **ssl-key-manager-provider**

  Specifies a key manager provider to use for the LDAP Server Extension. The key manager provider is not mandatory and can be used if the remote LDAP server is configured for client authentication. The referenced key manager provider must be enabled. For more information about key manager providers, see Configuring Key Manager Providers.

- **ssl-trust-all**

  If this parameter is set to `true`, all remote LDAP servers are trusted. The default value is `false`. Setting this value to `true` avoids having to import certificates from remote LDAP servers but is insecure.

> **Note:**
>
> Although the interactive `dsconfig --advanced` command offers Blind Trust as a possible trust manager provider, Blind Trust is not supported for the proxy server. Instead, if you want to avoid the import of certificates, set the `ssl-trust-all` parameter to **`true`**. This presents an insecure deployment and is not recommended for production environments, only for testing purposes.

If the `remote-ldap-server-ssl-policy` is set to `never`, then the value of the `ssl-trust-all` parameter is irrelevant. All connections between the proxy will be insecure (unencrypted) in this case. For more information on the `remote-ldap-server-ssl-policy`, see Understanding the Modes of Secure Connection.

- **ssl-trust-manager-provider**

  Specifies which trust manager provider to use for the LDAP Server Extension. The trust manager provider is mandatory unless the `ssl-trust-all` parameter is set to `true`. The referenced trust manager provider must be enabled.

- **ssl-protocol**

  Governs the SSL/TLS protocol that would be used during SSL communication with the remote LDAP server. This property takes system default values and can be overridden with valid SSL/TLS protocols as required. See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to understand about system default values.

- **ssl-cipher-suite**

  Governs the SSL/TLS cipher suite that would be used during SSL communication with the remote LDAP server. This property takes system default values and can be overridden with valid SSL/TLS cipher suites as required. See Supported TLS Protocols and Cipher Suites by Oracle Unified Directory to understand about system default values.

28

Controlling Access To Data

Controlling access to directory contents is an integral part of creating a secure directory service. Access to data is managed with access control instructions (ACIs) that specify the access right to individual entries, all sub-entries below an entry, or all entries on a global basis. Numerous or complicated ACIs require greater processing resources than a few simple ACIs. You can significantly reduce the performance of your directory by specifying a large number of ACIs or extremely complicated ACIs.

Oracle Unified Directory includes the ability to view the effective rights of a given user for a given entry. This feature simplifies the administration of the complex and powerful access control mechanism.

> **Note:**
>
> For an overview of the ACI model, see Understanding Access Control Model in Oracle Unified Directory.

Topics:

28.1 Managing Global ACIs Using `dsconfig`

Global ACIs control access to the root of the DIT instead of to a particular sub-tree. Global ACIs apply to all entries in the directory.

You can set, reset, and delete global ACIs with the `dsconfig` command and with the `ldapmodify` command. `dsconfig` accesses the server configuration over SSL, using the administration connector. For more information about `dsconfig` and managing non-global ACI's see the following topics:

- Adding a Global ACI

28.1.1 About Default Global ACIs

When you install Oracle Unified Directory, nine default global ACIs are defined. Review this topic for the effect of all the default global ACIs.

The effect of all the default global ACIs is to allow the following:

- Anyone has read access to certain controls and extended operations.
- Anyone has access to search, compare, and read attributes at the rootDSE level. Certain attributes require explicit access.
- Authenticated users can modify a subset of the attributes in their own entries in the directory. Users are unable to delete their own entries.
- Anyone has access to key operational attributes including many in the root DSE and `cn=schema`, as well as other attributes that show up in entries throughout the server.

The proxy forwards LDAP requests to the remote LDAP server, and then the remote LDAP server evaluates the ACIs. Hence, applicable ACIs for the identity used in the proxy bind request need to be configured in the remote LDAP server.

28.1.2 Displaying the Global ACIs

The global ACIs are all values of the `global-aci` property of the access control handler. You can use `dsconfig` to display the global ACIs currently configured on the server by viewing the `global-aci` property.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 5444 -D cn="Directory Manager" -j pwd-file.txt -X -n get-
access-control-handler-prop --property global-aci
Property   : Value(s)
-----------:---------------------------------------------------------------------
global-aci : (extop="1.3.6.1.4.1.26027.1.6.1 || 1.3.6.1.4.1.26027.1.6.3 ||
           : 1.3.6.1.4.1.4203.1.11.1 || 1.3.6.1.4.1.1466.20037 ||
           : 1.3.6.1.4.1.4203.1.11.3") (version 3.0; acl "Anonymous extended
           : operation access"; allow(read) userdn="ldap:///anyone";),
           : "(target="ldap:///")(targetscope="base")(targetattr="objectClass||
           : namingContexts||supportedAuthPasswordSchemes||supportedControl||su
           : pportedExtension||supportedFeatures||supportedLDAPVersion||support
           : edSASLMechanisms||vendorName||vendorVersion")(version 3.0; acl
           : "User-Visible Root DSE Operational Attributes"; allow
           : (read,search,compare) userdn="ldap:///anyone";)",
           : (target="ldap:///cn=changelog")(targetattr="*")(version 3.0; acl
           : "External changelog access"; deny (all) userdn="ldap:///anyone";),
           : "(target="ldap:///cn=schema")(targetscope="base")(targetattr="obje
           : ctClass||attributeTypes||dITContentRules||dITStructureRules||ldapS
           : yntaxes||matchingRules||matchingRuleUse||nameForms||objectClasses"
           : )(version 3.0; acl "User-Visible Schema Operational Attributes";
           : allow (read,search,compare) userdn="ldap:///anyone";)",
           : (target="ldap:///dc=replicationchanges")(targetattr="*")(version
           : 3.0; acl "Replication backend access"; deny (all)
           : userdn="ldap:///anyone";),
           : (targetattr="audio||authPassword||description||displayName||givenN
           : ame||homePhone||homePostalAddress||initials||jpegPhoto||labeledURI
           : ||mobile||pager||postalAddress||postalCode||preferredLanguage||tel
           : ephoneNumber||userPassword")(version 3.0; acl "Self entry
           : modification"; allow (write) userdn="ldap:///self";),
```

```
: "(targetattr="createTimestamp||creatorsName||modifiersName||modify
: Timestamp||entryDN||entryUUID||subschemaSubentry||orclguid||nsuniq
: ueid")(version 3.0; acl "User-Visible Operational Attributes";
: allow (read,search,compare) userdn="ldap:///anyone";)",
: "(targetattr="userPassword||authPassword")(version 3.0; acl "Self
: entry read"; allow (read,search,compare) userdn="ldap:///self";)",
: (targetcontrol="1.3.6.1.1.12 || 1.3.6.1.1.13.1 || 1.3.6.1.1.13.2
: || 1.2.840.113556.1.4.319 || 1.2.826.0.1.3344810.2.3 ||
: 2.16.840.1.113730.3.4.18 || 2.16.840.1.113730.3.4.9 ||
: 1.2.840.113556.1.4.473 || 1.3.6.1.4.1.42.2.27.9.5.9") (version
: 3.0; acl "Authenticated users control access"; allow(read)
: userdn="ldap:///all";), (targetcontrol="2.16.840.1.113730.3.4.2 ||
: 2.16.840.1.113730.3.4.17 || 2.16.840.1.113730.3.4.19 ||
: 1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 ||
: 2.16.840.1.113730.3.4.16 || 2.16.840.1.113894.1.8.31") (version
: 3.0; acl "Anonymous control access"; allow(read)
: userdn="ldap:///anyone";)
```

28.1.3 Deleting a Global ACI

The easiest way to delete a global ACI is to use `dsconfig` in interactive mode. Interactive mode walks you through the ACI configuration, and is therefore not documented here. If you delete global ACIs in non-interactive mode, then ensure that you escape all special characters in the ACI specification as required by your command line shell.

Let us assume that a customer had granted an anonymous ACI for all user attributes except for the `userpassword` and `authPassword` attributes previously. This example deletes that same global ACI by using `dsconfig` in non-interactive mode.

Run the `dsconfig` command as follows.

```
dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j /tmp/passwd.txt -X -n  \
set-access-control-handler-prop \
--remove global-aci:"(targetattr!=\"userPassword || authPassword\") \
(version 3.0 ; acl \" Anonymous read access\" ; allow ( read,search,compare ) \
userdn=\"ldap:///anyone\" ;)"
```

28.1.4 Adding a Global ACI

When you add a global ACI, ensure that you escape all special characters in the ACI specification as required by your command-line shell.

This example adds the global ACI that was removed in the previous procedure, using `dsconfig` in non-interactive mode.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D cn="Directory Manager" -j /tmp/passwd.txt -X -n \
set-access-control-handler-prop \
--add global-aci:"(targetattr="createTimestamp||creatorsName||modifiersName||
modifyTimestamp||entryDN||entryUUID||subschemaSubentry||orclguid||nsuniqueid") \
(version 3.0; acl \"User-Visible Operational Attributes\"; allow (read,search,compare) \
userdn=\"ldap:///anyone\";)"
```

28.2 Managing ACIs With `ldapmodify`

You can create access control instructions (ACIs) manually using LDIF statements, and add them to your directory by using the `ldapmodify` command. Because ACI values can be very complex, it is useful to view existing values and copy them to help create new ones.

For additional sample ACIs to the ones illustrated here, see Managing Access Control.

Tasks performed using ACIs using LDIF statements are explained in the following sections:

- Viewing ACI Attribute Values
- Adding an ACI
- Removing an ACI

28.2.1 Viewing ACI Attribute Values

ACIs are stored as one or more values of the `aci` attribute on an entry. The `aci` attribute is a multivalued operational attribute that can be read and modified by directory users, and should itself be protected by ACIs.

Administrative users are usually given full access to the `aci` attribute.

View the values of the `aci` attribute by running the following `ldapsearch` command:

```
$ ldapsearch -h host -p port -D "cn=Directory Manager" -j pwd-file \
  -b entryDN -s base "(objectclass=*)" aci
```

The result is LDIF text that you can copy into a new LDIF ACI definition for editing. Because the value of an ACI is a long string, the output from the `ldapsearch` operation is likely to be displayed over several lines, with the first space being a continuation marker. Take this into account when copying and pasting the LDIF output.

To view the effect of an ACI value, in terms of the permissions that it grants or denies, see Viewing Effective Rights.

28.2.2 Adding an ACI

You can add an ACI by specifying the ACI in an LDIF file and then applying the LDIF file with the `ldapmodify` command. The LDIF file must contain one or more `aci` attributes, each of which is composed of the `aci:` prefix followed by the ACI specification.

For more information, see Understanding the Syntax of Access Control Instructions.

To add an ACI:

1. Create the ACI in an LDIF file.

   The following sample LDIF file (`aci.ldif`) adds an ACI that grants a particular user (`csmith`) full access rights to the directory:

   ```
   dn: ou=people,dc=example,dc=com
   changetype: modify
   add: aci
   aci: (targetattr="*")(version 3.0; acl "give csmith full rights"; allow(all)
   userdn =  "ldap:///uid=csmith,ou=People,dc=example,dc=com";)
   ```

2. Use the `ldapmodify` command to apply the ACI to the directory.

   The following command applies the ACI contained in the `aci.ldif` file to the directory:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
     --filename aci.ldif
   Processing MODIFY request for ou=people,dc=example,dc=com
   MODIFY operation successful for DN ou=people,dc=example,dc=com
   ```

28.2.3 Removing an ACI

You can remove an ACI by specifying its value in an LDIF file, and then removing the value with the `ldapmodify` command.

To remove an ACI:

1. Remove the ACI in an LDIF file.

   The following sample LDIF file (`remove-aci.ldif`) removes the ACI that was added in the previous procedure:

   ```
   dn: ou=people,dc=example,dc=com
   changetype: modify
   delete: aci
   aci: (targetattr="*")(version 3.0; acl "give csmith full rights"; allow(all)
   userdn =  "ldap:///uid=csmith,ou=People,dc=example,dc=com";)
   ```

2. Use the `ldapmodify` command to apply the change to the directory.

   The following command applies the changes contained in the `remove-aci.ldif` file to the directory:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
   --filename remove-aci.ldif
   Processing MODIFY request for ou=people,dc=example,dc=com
   MODIFY operation successful for DN ou=people,dc=example,dc=com
   ```

28.3 Managing Access Control Using OUDSM

You can use OUDSM to view the existing ACIs that are configured in the server, to create new access control points, and to create new ACIs in a user-friendly interface.

Managing access control by using OUDSM is described in the following sections:

- Displaying the Configured ACIs
- Creating an Access Control Point
- Creating an Access Control Point Based on an Existing Access Control Point
- Deleting an Access Control Point.
- Adding an ACI
- Adding an ACI Based on an Existing ACI
- Modifying an ACI

28.3.1 Displaying the Configured ACIs

Oracle Unified Directory supports several preconfigured access control instructions (ACIs), by default. You can use Oracle Unified Directory Services Manager (OUDSM) to display all ACIs that are configured in the server.

To display all ACIs that are configured in the server by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.

3. Expand the **Directory ACLs** element.
4. All configured ACIs are listed under the access control point in which the ACI is defined. Expand the access control point to view the ACIs. For example, to display the list of ACIs that apply to the Root entry, expand the Root entry.
5. Select an ACI to view its properties in the right hand pane.

28.3.2 Creating an Access Control Point

An access control point is the entry in which an ACI is defined, in other words, the entry that contains the corresponding `aci` attribute.

To define a new access control point by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Click the **Create** icon.
5. In the **Location** field, enter the DN of the entry that will be the new access control point, or click **Select** to select the entry from the directory.
6. To add one or more ACIs to the access control point, click **Create ACI**.
7. Enter the ACI details. For more information about these fields, see Adding an ACI.
8. When you have added the required ACIs to the access control point, click **Create**.

28.3.3 Creating an Access Control Point Based on an Existing Access Control Point

Use Oracle Unified Directory Services Manager (OUDSM) to define a new access control point that is based on an existing access control point.

To define a new access control point that is based on an existing access control point by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Select the access control point on which you want to base the new access control point.
5. Click the **Create like** icon.
6. In the **Location** field, enter the DN of the entry that will be the new access control point, or click **Select** to select the entry from the directory.
7. The new access control point is automatically created with the same ACL as the access control point on which it was based.
8. To add, remove, or edit the existing ACIs on the new access control point, click **Create**, **Edit** or **Delete**.
9. To add or edit an ACI, enter the required details. For more information about these fields, see Adding an ACI.

10. When you have modified the ACIs for the new access control point, click **Create**.

28.3.4 Deleting an Access Control Point

Use Oracle Unified Directory Services Manager to delete an access control point.

To delete an access control point by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Select the access control point that you want to delete and click the **Delete** icon.
5. Click **OK** to confirm the deletion.

28.3.5 Adding an ACI

Use Oracle Unified Directory Services Manager (OUDSM) to add an access control instruction (ACI) to an existing access control point.

To add an ACI to an existing access control point by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Expand the access control point to which you want to add the new ACI.
5. Select one of the ACIs in the access control list.
6. Click the **Add** icon.
7. To build the ACI in a user friendly interface, select the **Detail View** tab.
8. Select the **Scope** of the ACI.

   Usually an ACI has subtree scope. You can restrict the scope of the ACI by selecting one of the following values:

   - **Base.** The ACI applies to the target resource only.
   - **One.** The ACI applies to the target resource's first-generation children.
   - **Subtree.** The ACI applies to the target resource and the subtree below it.
   - **Subordinate.** The ACI applies only to the subtree below the target resource.
9. In the **Targets** field, select each element of the ACI and click **Edit** to define its properties.

   For more information about defining ACI targets, see Defining Targets.

   You can now target one or more attributes that occur in the targeted entries to deny or allow access to partial information about an entry, by performing the following steps:

   a. In the **Targets** field, select **Target Attribute** and click **Edit**.
   b. For **Operator**, select the desired value.
   c. For **Attributes**, select the desired option.

d. Click **Add** to enter the one or more ACI Attributes and subtypes. You can also click **Search** to search for the attribute name.

   You can enter subtypes for the attributes in the **sub-type (optional)** field. You can enter multiple subtypes for same attribute.

e. Click **OK**.

For more information, see Targeting Attributes in a Targeted Entry.

10. In the **Permissions** field, click the **Add** icon to define permissions and bind rules.

    For more information about defining ACI permissions, see Setting Permissions.

    For more information about defining bind rules, see Understanding Bind Rules.

    To define the bind rules:

    a. From **Bind Rule Type** list, select the desired bind rule.

    b. Click the **User Attribute** tab to create user attribute bind rule.

    c. For **User Attribute Operator** property, select the desired value.

    d. For **Entry Selection** property, select **Target Entry and its Subtree.**

    e. From the **Inheritance Levels** list, select the desired inheritance level value.

    f. In the **User Attribute** field, enter an attribute or alternatively click **Select** to search an entry.

    g. For **User Attribute Type** property, click **Bind Type Format.**

    h. From the **Bind Type Value** list, select the bind type value.

    i. Click **OK.**

11. If you would rather define the ACI manually, click the **Text Editor View** tab and enter the details of the ACI.

    Click **Validate** to check that the ACI conforms to the ACI syntax.

    You can also use this view to copy and paste existing ACIs.

12. When you have completed the ACI definition, click *Create*.

28.3.6 Adding an ACI Based on an Existing ACI

Use Oracle Unified Directory Services Manager (OUDSM) to add an access control instruction (ACI) that is based on an existing ACI.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Expand the access control point that contains the ACI that you want to copy.
5. Select the ACI that you want to copy.
6. Click the **Add like** icon.
7. Edit the elements of the ACI that you want to change, either in **Text Editor View** or in **Detail View**.
8. When you have completed the ACI definition, click **Create**.

28.3.7 Modifying an ACI

Use Oracle Unified Directory Services Manager (OUDSM) to modify an existing access control instruction (ACI).

To modify an existing ACI by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Directory ACLs** element.
4. Expand the access control point that contains the ACI that you want to change
5. Select the ACI that you want to change.
6. Edit the elements of the ACI, either in **Text Editor View** or in **Detail View**.
7. When you have completed your changes, click **Apply**.

28.4 Managing Macro ACIs Using OUDSM

You can use OUDSM to enter macro expressions in `target`, `targetFilter`, `userDn`, `groupDN`, and `userAttr` attributes.

For more information about Macro ACIs, see Using Macro ACIs for Advanced Access Control.

This section contains the following topics:

- Editing a Target
- Editing a Target Filter
- Editing Bind Rules for User DN or Group DN
- Editing Bind Rules for User Attributes

28.4.1 Editing a Target

Use Oracle Unified Directory Services Manager to edit a target to enter a macro access control instruction.

To edit a target to enter a macro ACI:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. From the **Directory ACLs** list, select the ACI that you want to edit.
4. From the Targets table, select the **Target** row.
5. Click **Edit.**
6. In the **Target** field, enter the macro expression.
7. Click **OK.**

28.4.2 Editing a Target Filter

Use Oracle Unified Directory Services Manager to edit a target filter.

To edit a target filter:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. From the **Directory ACLs** list, select the ACI that you want to edit.
4. From the Targets table, select the **Target Filter** row**.**
5. Click **Edit.**
6. In the **Target** field, enter the filter with the macro expression.
7. Click **OK.**

28.4.3 Editing Bind Rules for User DN or Group DN

Use Oracle Unified Directory Services Manager to define access for a targeted resource to a specific user or a specific group.

To edit bind rules for user DN or group DN:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. From the **Directory ACLs** list, select the ACI that you want to edit.
4. From the Permissions table, select the **Bind Rules** row.
5. Click **Edit.**
6. From Bind Rule Type list, select the desired bind rule.
7. On the **Access To** tab, from the **User DN** list, select **Specify Users.**
8. Perform the following steps in the User table:
    a. Click **Add.**
    b. Enter the macro expression to define user access or alternatively click **Select** to search the entry and add the macro expression in the selected entry.
9. Perform the following steps to specify access to a specific group for a targeted resource in the Group DN Operator table:
    a. Click **Add.**
    b. Enter the macro expression to define group access or alternatively click **Select** to search the entry and add the macro expression in the selected entry.
10. Click **OK.**

> **Note:**
>
> You can edit an individual bind rule as well. You must select the required bind rule in the Permissions table, and then click **Edit** for modifying the bind rule.

28.4.4 Editing Bind Rules for User Attributes

Use Oracle Unified Directory Services Manager (OUDSM) to edit bind rules for user attributes.

To edit bind rules for user attributes:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. From the **Directory ACLs** list, select the ACI that you want to edit.
4. From the Permissions table, select the **Bind Rules** row.
5. Click **Edit.**
6. Click the **User Attribute** tab.
7. From Bind Rule Type list, select the desired bind rule.
8. For Entry Selection property, select **Specific Entry.**
9. In the Entry Base DN field, enter the base DN or alternatively click **Select** to search an entry and add the macro expression in the selected entry.
10. In the User Attribute field, enter an attribute name or alternatively click **Select** to search an attribute name from the list of attribute names.
11. Click **Bind Type Format.**
12. From the Bind Type Value list, select the desired value.
13. Click **OK.**

> **Note:**
>
> You can edit an individual bind rule as well. You must select the required bind rule in the Permissions table, and then click **Edit** for modifying the bind rule.

28.5 Managing Access Control

Review these topics for tasks related to access control point.

- Granting Write Access to Personal Entries
- Granting a Group Full Access to a Suffix
- Granting Rights to Add and Delete Group Entries
- Allowing Users to Add or Remove Themselves from a Group
- Granting Conditional Access to a Group

- Denying Access
- Defining Permissions for DNs that Contain a Comma

28.5.1 Granting Write Access to Personal Entries

The default global ACIs allow write access to a limited subset of the attributes of a user's own entry.

These attributes include the following:

- `audio`
- `authPassword`
- `description`
- `displayName`
- `givenName`
- `homePhone`
- `homePostalAddress`
- `initials`
- `jpegPhoto`
- `labeledURI`
- `mobile`
- `pager`
- `postalAddress`
- `postalCode`
- `preferredLanguage`
- `telephoneNumber`
- `userPassword`

The following topics provide the procedures to grant users write access to additional attributes of their own entries:

- Granting Write Access Based on DNS
- Granting Write Access Based on Authentication Method

28.5.1.1 Granting Write Access Based on DNS

The following example ACI enables users internal to `example.com` to change their own business category and room number.

Remember, by allowing write access, you also grant users the right to delete attribute values.

```
aci: (targetattr="businessCategory || roomNumber")
(version 3.0; acl "Write example.com"; allow (write)
userdn="ldap:///self" and dns="*.example.com";)
```

This example assumes that the ACI is added to the `ou=People,dc=example,dc=com` entry.

28.5.1.2 Granting Write Access Based on Authentication Method

The following example enables users to update all of their own personal information in the `example.com` tree if they establish an SSL connection to the directory.

By setting this permission, you are also granting users the right to delete attribute values.

```
aci: (targetattr="*")
(version 3.0; acl "Write SSL"; allow (write)
userdn= "ldap:///self" and authmethod="ssl";)
```

This example assumes that the `aci` is added to the `ou=subscribers,dc=example,dc=com` entry.

28.5.2 Granting a Group Full Access to a Suffix

Most directories have a group that is used to identify certain corporate functions. These groups can be given full access to all or part of the directory. By applying the access rights to the group, you can avoid setting the access rights for each member individually. Instead, you grant users these access rights by adding them to the group.

The following sample ACI allows a group named the `HRgroup` full access to the `ou=People` branch of the directory so that they can update employee information:

```
aci: (targetattr="*") (version 3.0; acl "HR"; allow (all)
groupdn= "ldap:///cn=HRgroup,ou=People,dc=example,dc=com";)
```

This example assumes that the ACI is added to the `ou=People,dc=example,dc=com` entry.

28.5.3 Granting Rights to Add and Delete Group Entries

Some organizations want to allow employees to create entries in the tree if it can increase their efficiency, or if it can contribute to the corporate dynamics.

The following examples assume that `example.com` has a social committee that is organized into various clubs (tennis, swimming, skiing, and so on):

- Creating a `"Create Group"` ACI
- Creating a `"Delete Group"` ACI

28.5.3.1 Creating a `"Create Group"` ACI

This sample ACI allows any `example.com` employee to create a group entry representing a new club, under the `ou=social committee` branch.

```
aci: (target = "ldap:///dc=ou=social committee,dc=example,dc=com")
(targetfilter="(|(objectClass=groupOfNames)(objectClass=top))")
(version 3.0; acl"Create Group"; allow (search,read,add) (userdn =
@ "ldap:///uid=*,ou=People,dc=example,dc=com" and dns = "*.example.com");)
```

This example assumes that the ACI is added to the `ou=social committee, dc=example,dc=com` entry.

> **Note:**
>
> This ACI does not grant write permission, which means that the entry creator cannot modify the entry. Because the server adds the value `top` behind the scenes, you must specify `objectClass=top` in the `targattrfilters`.

28.5.3.2 Creating a `"Delete Group"` ACI

This sample ACI ensures that only the group owner can modify or delete a group entry under the ou=Social Committee branch.

```
aci: (target="ou=social committee,dc=example,dc=com")
(targetattr = "*")
(targattrfilters="del=objectClass:(objectClass=groupOfNames)")
(version 3.0; acl "Delete Group"; allow (write,delete)
userattr="owner#GROUPDN";)
```

This example assumes that the ACI is added to the `ou=social committee,dc=example,dc=com` entry.

28.5.4 Allowing Users to Add or Remove Themselves from a Group

Many directories set ACIs that allow users to add or remove themselves from groups. This is useful, for example, for allowing users to add and remove themselves from mailing lists.

The following sample ACI enables all employees to add themselves to any group entry under the `ou=social committee` subtree:

```
aci: (targettattr="member")(version 3.0; acl "Group Members";
allow (selfwrite)
(userdn= "ldap:///uid=*,ou=People,dc=example,dc=com") ;)
```

This example assumes that the ACI is added to the `ou=social committee, dc=example,dc=com` entry.

28.5.5 Granting Conditional Access to a Group

Usually, when you grant a group privileged access to the directory, you want to ensure that those privileges are protected from intruders trying to impersonate the privileged users. Therefore, access control rules that grant critical access to a group or role are often associated with several conditions.

The following sample ACI grants the Directory Administrators group full access to the corporate clients branch of the directory tree, provided the following conditions are fulfilled:

- The connection is authenticated using a certificate over SSL
- Access is requested between 08:00 and 18:00, Monday through Thursday
- Access is requested from a specified IP address

```
aci: (target="ou=corporate-clients,dc=example,dc=com")
(targetattr = "*") (version 3.0; acl "corporate-clients"; allow (all)
(groupdn="ldap:///cn=DirectoryAdmin,ou=corporate-clients,dc=example,dc=com")
and (authmethod="ssl") and (dayofweek="Mon,Tue,Wed,Thu") and
(timeofday >= "0800" and timeofday <= "1800") and (ip="255.255.123.234"); )
```

This example assumes that the ACI is added to the `ou=corporate-clients,dc=example,dc=com` entry.

28.5.6 Denying Access

If your directory holds business-critical information, you might specifically want to deny access to it. The following sample ACIs allow users to read certain "billing information", such as connection time and account balance, under their own entries, but prohibits them from changing this information.

This ACI allows users to read the information. The example assumes that the relevant attributes have been created in the schema.

```
aci: (targetattr="connectionTime || accountBalance")
(version 3.0; acl "Billing Info Read"; allow (search,read)
userdn="ldap:///self";)
```

This ACI prevents users from changing the information. The example assumes that the relevant attributes have been created in the schema.

```
aci: (targetattr="connectionTime || accountBalance")
(version 3.0; acl "Billing Info Deny";
deny (write) userdn="ldap:///self";)
```

28.5.7 Defining Permissions for DNs that Contain a Comma

DNs that contain commas require special treatment within LDIF ACI statements. In the target and bind rule portions of the ACI statement, commas must be escaped by a single backslash (\).

The following example illustrates this syntax:

```
dn: o=example.com Bolivia\, S.A.
objectClass: top
objectClass: organization
aci: (target="ldap:///o=example.com Bolivia\,S.A.")
(targetattr="*") (version 3.0; acl "aci 2"; allow (all)
groupdn = "ldap:///cn=Directory Administrators,
o=example.com Bolivia\, S.A.";)
```

28.6 About Proxy Authorization ACIs

The proxy authorization method is a special form of authentication: a user that binds to the directory using his own identity is granted the rights of another user, through proxy authorization.

This example makes the following assumptions:

- The client application's bind DN is `uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com`.
- The targeted subtree to which the client application is requesting access is `ou=Accounting,dc=example,dc=com`.
- An Accounting Administrator with access permissions to the `ou=Accounting,dc=example,dc=com` subtree exists in the directory.

For the client application to gain access to the Accounting subtree (using the same access permissions as the Accounting Administrator), the application requires the following rights and controls:

- The Accounting Administrator must have access permissions to the `ou=Accounting,dc=example,dc=com` subtree. The following ACI grants all rights to the Accounting Administrator entry:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
(targetattr="*") (version 3.0; acl "allow All-AcctAdmin"; allow
(all) userdn="ldap:///uid=AcctAdministrator,ou=Administrators,
dc=example,dc=com";)
```

- The client application must have proxy rights. The following ACI grants proxy rights to the client application:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
(targetattr="*") (version 3.0; acl "allow proxy-
accounting software"; allow (proxy) userdn=
"ldap:///uid=MoneyWizAcctSoftware,ou=Applications,
dc=example,dc=com";)
```

- The client application must be allowed to use the proxy authorization control. The following ACI allows the client application to use the proxy authorization control:

```
aci: (targetcontrol = "2.16.840.1.113730.3.4.18")
(version 3.0; acl "allow proxy auth - accounting software";
allow (all) userdn="ldap:///uid=MoneyWizAcctSoftware,ou=Applications,
dc=example,dc=com";)
```

With these ACIs in place, the `MoneyWizAcctSoftware` client application can bind to the directory and send an LDAP command such as `ldapsearch` or `ldapmodify` that requires the access rights of the proxy DN.

In the previous example, if the client wanted to perform an `ldapsearch` command, the command would include the following controls:

```
$ ldapsearch -D "uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com" \
  -j pwd-file -Y "dn:uid=AcctAdministrator,ou=Administrators,dc=example,dc=com" \
  -b "ou=Accounting,dc=example,dc=com"  "objectclass=*"\
...
```

The base of the search must match the target of the ACIs. The client binds as itself but is granted the privileges of the proxy entry. The client does not need the password of the proxy entry.

For more information, see Searching Using the Proxied Authorization Control.

28.7 Viewing Effective Rights

When you maintain the access control policy on the entries of a directory, it is useful to know the effects on security of the ACIs that you define. The directory server enables you to evaluate existing ACIs and report the effective rights that they grant for a given user on a given entry.

The following topics provide a detailed information on effective rights:

28.7.1 About Get Effective Rights Control

The directory server responds to the Get Effective Rights control, which can be included in a search operation. The response to this control is to return the effective rights information about the entries and attributes in the search results. This extra information includes read and write permissions for each entry and for each attribute in each entry.

The permissions may be requested for the bind DN used for the search or for an arbitrary DN, allowing administrators to test the permissions of directory users.

Effective rights functionality relies on an LDAP control. To view the effective rights when going through a proxy server, you must enable this control in the proxy chaining policy. You must also ensure that the proxy identity used to bind to the remote server is also allowed to access the effective rights attributes.

28.7.2 Using the Get Effective Rights Control

The behavior of the Get Effective Rights Control differs from the Internet draft Get Effective Rights Control. Review this topic to note the different ways.

For more information on the Internet draft Get Effective Rights Control, see `http://tools.ietf.org/html/draft-ietf-ldapext-acl-model-08`.

- There is no response control returned with the search results. Instead, the rights information is added to the result entries. Also, the format of the rights information is completely different from the draft and is described below.
- The request control only takes an `authzid`.

There are two ways to specify the Get Effective Rights control with the `ldapsearch` command:

1. Use the `-J "1.3.6.1.4.1.42.2.27.9.5.2"` option or simply `-J effectiverights`. If you specify a NULL value for the Get Effective Rights Control's `authzid` value, the bind user is used as the `authzid` and the rights for the attributes and entries being returned with the current `ldapsearch` operation are retrieved.
2. The simpler and preferred method is to use the `-g` option with or without the `-e` option:
   - `-g "dn: DN"`--The search results will show the effective rights of the user binding with the given *DN*. This option allows an administrator to check the effective rights of another user. The option `-g "dn:"` will show the effective rights for anonymous authentication.
   - `-e attributeName1 -e attributeName2` --The search results will also include the effective rights on the named attributes. This option can be used to specify attributes that would not appear in the search results for the entry. For example, this option can be used to determine if a user has permission to add an attribute that does not currently exist in an entry.

> **Note:**
>
> The `-e` option requires the `-g` option and should not be used with the `-J` option.
>
> If you use the `-g` option, do not use the `-J` option with the OID of the Get Effective Rights control.

Besides using one of these two ways to specify the Get Effective Rights Control, you must specify the type of information you want to view, either the simple rights or the more detailed logging information that explains how those rights are granted or denied. The type of information is determined by adding either `aclRights` or `aclRightsInfo`, respectively, as an attribute to return in the search results. You can request both attributes to receive all effective rights information, although the simple rights are redundant with the information in the detailed logging information.

> **Note:**
>
> The `aclRights` and `aclRightsInfo` attributes have the behavior of virtual operational attributes. They are not stored in the directory, and they will not be returned unless explicitly requested. The directory server generates these attributes in response to the Get Effective Rights Control. For this reason, do not use either of these attributes in filters or search operations of any kind.

The effective rights feature inherits other parameters that affect access control (such as time of day, authentication method, machine address, and machine name) from the user initiating the search operation.

The following example shows how a user, Carla Fuente, can view her rights in the directory. In the results, a `1` means that permission is granted, and a `0` means that permission is denied.

```
$ ldapsearch -J effectiverights -h rousseau.example.com -p 1389 \
  -D "uid=cfuente,ou=People,dc=example,dc=com" -j pwd-file \
  -b "dc=example,dc=com" "(objectclass=*)" aclRights
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

This result shows Carla Fuente the entries in the directory where she has at least read permission and that she can modify her own entry. The effective rights control does not bypass normal access permissions, so a user will never see the entries for which they do not have read permission. In the following example, the Directory Manager can see the entries to which Carla Fuente does not have read permission:

```
$ ldapsearch -h rousseau.example.com -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -g "dn: uid=cfuente,ou=People,dc=example,dc=com" \
  -b "dc=example,dc=com" "(objectclass=*)" aclRights
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Directory Administrators, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=Special Users,dc=example,dc=com
```

```
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

In the output above, the directory manager can see that Carla Fuente cannot even view the Special Users nor the Directory Administrators branches of the directory tree. In the following example, the Directory Administrator can see that Carla Fuente cannot modify the `mail` and `manager` attributes in her own entry:

```
$ ldapsearch -h rousseau.example.com -p 1389 -D "cn=Directory Manager" \
  -j pwd-file -g "dn: uid=cfuente,ou=People,dc=example,dc=com" \
  -b "dc=example,dc=com" "(uid=cfuente)" aclRights "*"
version: 1
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;attributeLevel;mail: search:1,read:1,compare:1,
write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
mail: cfuente@example.com
aclRights;attributeLevel;uid: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
uid: cfuente
aclRights;attributeLevel;givenName: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
givenName: Carla
aclRights;attributeLevel;sn: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
sn: Fuente
aclRights;attributeLevel;cn: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
cn: Carla Fuente
aclRights;attributeLevel;userPassword: search:0,read:0,
compare:0,write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
userPassword: {SSHA}wnbWHIq2HPiY/5ECwe6MWBGx2KMiZ8JmjF80Ow==
aclRights;attributeLevel;manager: search:1,read:1,compare:1,
write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
manager: uid=bjensen,ou=People,dc=example,dc=com
aclRights;attributeLevel;telephoneNumber: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
telephoneNumber: (234) 555-7898
aclRights;attributeLevel;objectClass: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
```

28.7.3 Understanding Effective Rights Results

For every effective rights request, a result is generated depending on the options specified.

- Effective Rights Information
- `write`, `selfwrite_add`, and `selfwrite_delete` Permissions

- Effective Rights Logging Information

28.7.3.1 Effective Rights Information

The effective rights information is presented according to the following subtypes:

Table 28-1 Subtypes of effective rights information

Subtypes	Description
clRights;entrylevel	Presents entry-level rights information
aclRights;attributelevel	Presents attribute-level rights informationel
`aclRightsInfo;entrylevel`	Presents entry-level logging information
`aclRightsInfo;attributelevel`	Presents attribute-level logging information

The format of the `aclRights` string is as follows:

`aclRights;entryLevel:` *permission*:*value*(*permission*:*value*)*

and

`aclRights;attributeLevel:` *permission*:*value*(*permission*:*value*)*

The possible entry-level permissions are `add`, `delete`, `read`, `write`, and `proxy`. The possible values for each permission are `0` (permission not granted) and `1` (permission granted).

Entry-level Permission	Explanation
`add` and `delete`	The ability of a user to add and delete the entire entry.
`read`	The ability of a user to read and search attributes in the entry.
`write`	The ability of a user to add, delete, and replace attribute values in the entry.
`proxy`	The ability of a user to access the directory with the rights of the entry.

> **Note:**
>
> For information about assigning these permissions in an ACI, see Understanding the Syntax of Access Control Instructions.

The possible attribute-level permissions are `read`, `search`, `compare`, `write`, `selfwrite_add`, `selfwrite_delete`, and `proxy`. The possible values for each permission are `0` (permission not granted) and `1` (permission granted). For the case of the write permission, the value of "`?`" is also permitted.

Attribute-level Permission	Explanation
`read`	The ability of a user to read the attribute value in the entry.

Attribute-level Permission	Explanation
`search`	The ability of a user to search the attribute value in the entry.
`compare`	The ability of a user to compare the attribute value in the entry with a value that is provided by the user.
`write`	The ability of a user to add, delete, and replace the attribute value in the entry. This applies to all attributes except the `authorization dn`.
`selfwrite_add`	The ability of a user to add the attribute, `authorization dn`.
`selfwrite_delete`	The ability of a user to delete the attribute, `authorization dn`.
`proxy`	The ability of a user to access the directory with the rights of the attribute in the entry.

> **Note:**
>
> The `write`, `selfwrite_add`, and `selfwrite_delete` permissions are particularly complex. If you see a "`?`", consult the logging information to establish why the permissions will or will not be granted. For more information, see Table 28-2.

The format of the `aclRightsInfo` string is as follows:

```
aclRightsInfo;logs;entryLevel;permission: acl_summary(main):permission-string
```

and

```
aclRightsInfo;logs;attributeLevel;permission;attribute:
acl_summary(main):permission-string
```

The entry-level and attribute-level permissions are described in the preceding section.

The *permission-string* contains detailed information about the effective rights at the entry-level and attribute-levels.

28.7.3.2 `write`, `selfwrite_add`, and `selfwrite_delete` Permissions

The attribute-level permission for `write` can be either `0`, `1`, or "`?`". Only `write` attribute-level permissions can have a value of "`?`", which usually results from a `targattrfilters` ACI component. For add and delete permissions, the entries that can be modified depend on the values of the attributes in the entry. Only the permission, `0` or `1`, is returned on the entries as they are returned with the `ldapsearch` operation.

For all attribute values except the `authorization dn`, if the value for a `write` permission is `1`, the permission is granted for both add and delete. Similarly, for all attribute values except the `authorization dn`, a value of `0` for a write permission means that the permission is not granted for either add or delete `ldapmodify` operations. The permission in force for the value of the `authorization dn` is returned explicitly in one of the `selfwrite` permissions, that is, either `selfwrite_add` or `selfwrite_delete`.

Although `selfwrite_add` and `selfwrite_delete` attribute-level permissions do not exist in the context of ACIs, a set of ACIs can grant a user `selfwrite` permission for just the add or just the delete part of a modify operation. For `selfwrite` permissions, the value of the attribute being modified is the `authorization dn`. The same distinction is not made for `write` permissions because the value of the attribute being modified for a write permission is undefined.

When the effective permission depends on a `targattrfilters` ACI, the "`?`" value indicates that the logging information should be consulted for more permission detail. The interdependencies between the `write`, `selfwrite_add`, and `selfwrite_delete` permissions are fairly complex and are outlined in the following table.

Table 28-2 Effective Rights Permission Interdependencies

write	selfwrite_add	selfwrite_delete	Effective Rights Explanation
0	0	0	Cannot add or delete any values of this attribute.
0	0	1	Can only delete the value of the `authorization dn`.
0	1	0	Can only add the value of the `authorization dn`.
0	1	1	Can only add or delete the value of the `authorization dn`.
1	0	0	Can add or delete all values except the `authorization dn`.
1	0	1	Can delete all values including the `authorization dn` and can add all values excluding the authorization dn.
1	1	0	Can add all values including the `authorization dn` and can delete all values excluding the `authorization dn`.
1	1	1	Can add or delete all values of this attribute.
?	0	0	Cannot add or delete the `authorization dn` value, but might be able to add or delete other values. See logging information for further details regarding the `write` permission.
?	0	1	Can delete but cannot add the value of the `authorization dn`, and might be able to add or delete other values. See logging information for further details regarding the `write` permission.
?	1	0	Can add but cannot delete the value of the `authorization dn` and might be able to add or delete other values. See logging information for further details regarding the `write` permission.
1	?	1	Can add and delete the value of the `authorization dn` and might be able to modify add, modify, or delete other values. See logging information for further details regarding the `write` permission.

28.7.3.3 Effective Rights Logging Information

The effective rights logging information enables you to understand and debug access control difficulties. The logging information contains an access control summary statement, called the `acl_summary`, that indicates why access control has been allowed or denied. The access control summary statement includes the following information:

- Whether access was allowed or denied
- The permissions granted
- The target entry of the permissions
- The name of the target attribute
- The subject of the rights being requested
- Whether the request was made by proxy, and if so, the proxy authentication DN
- The reason for allowing or denying access (important for debugging purposes as explained in the following table)

The following table lists the effective rights logging information reasons and their explanations.

Table 28-3 Effective Rights Logging Information Reasons and Their Explanations

Logging Information Reason	Explanation
`no reason available`	No reason available to explain why access was allowed or denied.
`no allow acis`	No `allow` ACIs exist, which results in denied access.
`result cached deny`	Cached information was used to determine the access denied decision.
`result cached allow`	Cached information was used to determine the access allowed decision.
`evaluated allow`	An ACI was evaluated to determine the access allowed decision. The name of the ACI is included in the log information.
`evaluated deny`	An ACI was evaluated to determine the access denied decision. The name of the ACI is included in the log information.
`no acis matched the resource`	No ACIs match the resource or target, which results in denied access.
`no acis matched the subject`	No ACIs match the subject requesting access control, which results in denied access.
`allow anyone aci matched anon user`	An ACI with a `userdn = "ldap:///anyone"` subject allowed access to the anonymous user.
`no matching anyone aci for anon user`	No ACI with a `userdn= "ldap:///anyone"` subject was found, so access for the anonymous user was denied.
`user root`	The user is root DN and is allowed access.

> **Note:**
>
> Write permissions for virtual attributes are not provided, nor is any associated logging evaluation information, because virtual attributes cannot be updated.

28.7.4 Restricting Access to the Get Effective Rights Control

Viewing effective rights is itself a directory operation that should be protected and appropriately restricted. The default ACI does not allow read access to the `aclRights` and `aclRightsInfo` operational attributes used to return effective rights. Create a new ACI for these attributes to enable access by directory users to this information.

For example, the following ACI allows members of the Directory Administrators group to get effective rights:

```
aci: (targetattr = "aclRights||aclRightsInfo")(version 3.0; acl "getEffectiveRights";
allow(all) groupdn = "ldap:///cn=Directory
Administrators,ou=Groups,dc=example,dc=com";)
```

In addition, access is needed to use the Get Effective Rights Control.

To enable access by directory users to the Get Effective Rights Control, create a new ACI target by using the OID (`1.3.6.1.4.1.42.2.27.9.5.2`) for this control. For additional ACI syntax information, see Defining Targets.

For example, the following ACI allows members of the Directory Administrators group to use the Get Effective Rights control:

```
aci: (targetcontrol = "1.3.6.1.4.1.42.2.27.9.5.2")(version 3.0;
acl "getEffectiveRights control access";
allow(all) groupdn = "ldap:///cn=Directory
Administrators,ou=Groups,dc=example,dc=com";)
```

29
Managing Administrative Users

You can create and change the password and privileges of a root user. You can also view and create administrators with limited privileges.
Topics:

29.1 About Privilege Subsystem

Oracle Unified Directory provides a flexible Privilege Subsystem that allows you to configure root users, Global Administrators, and administrators for your server. You can configure multiple root users and assign different root privileges to each administrator. For administrative domains, you can also configure multiple Global Administrators to manage administrative domains in your network or in a replicated environment.

Before you start using the procedures provided in this chapter, you must determine the following guidelines for your server:

- Number of root users, their privileges, and resource limits, if any.
- Number of administrators, their privileges, and resource limits, if any.
- Guidelines for user accounts on your system.
- Password policies for the server and for specific groups of users.

29.2 Defining Root Users

Review these topics for descriptive information about root users and the privilege subsystem.

29.2.1 About Root User

Oracle Unified Directory provides one default root DN or root user, `"cn=Directory Manager"`. The default root DN is a user entry assigned with specialized privileges including full read and write access to all data in the server.

Comparable to a UNIX root user or superuser, the root DN can bypass access controls to perform tasks on the server. The root user is defined below the `"cn=Root DNs,cn=config"` branch of the server at `cn=Directory Manager,cn=Root DNs,cn=config`.

Root users differ from regular user entries in the following ways:

- **Configuration**. Root users are the only user accounts that can exist in the server configuration (`cn=config`).
- **Privilege inheritance**. Root users automatically inherit the set of default root user privileges. Regular users do not automatically receive any privileges unless explicitly granted. You can grant privileges using real, virtual root-privilege-name attributes, or both in the entry.
- **Lockdown mode**. Root users are the *only* users who can cause the server to enter or leave lockdown mode, and only over the loopback interface.

29.2.2 About Multiple Root Users

The server supports multiple root users who have their own entries and their own set of credentials on the server. This allows you to assign privileges to a user who might need root access for a particular task, but might not need the full set of root user privileges.

With each entry, you can assign strong authentication such as the GSSAPI SASL mechanism, password policies, or add resource limits (if your schema allows it) to one root user while having a completely different configuration for another root user.

The Privilege Subsystem supports the configuration of multiple root users.

29.2.3 Root Users and the Privilege Subsystem

The Privilege Subsystem allows you to assign refined privileges to users who might require only a specific set of root user access privileges. Root users are automatically granted a set of privileges defined in the `default-root-privilege-name` attribute in the `"cn=Root DNs,cn=config"` subtree.

The Privilege Subsystem is independent from the Access Control Subsystem, but some operations might be subject to access controls.

The following table lists a set of privileges that are automatically assigned to the root user.

Privilege	Description
`backend-backup`	Allows the user to request the back-end backup task.
`backend-restore`	Allows the user to request the back-end restore task.
`bypass-acl`	Allows the user to bypass access control evaluation.
`bypass-lockdown`	Allows the associated user to bypass server lockdown mode.
`cancel-request`	Allows the user to cancel arbitrary client requests.
`config-read`	Allows the user to have read access to the server configuration.
`config-write`	Allows the user to have write access to the server configuration.
`disconnect-client`	Allows the user to terminate arbitrary client connections.
`ldif-export`	Allows the user to request the LDIF export task.
`ldif-import`	Allows the user to request the LDIF import task.
`modify-acl`	Allows the user to make changes to access control instructions defined in the server.
`password-reset`	Allows the user to reset the user passwords.

Privilege	Description
privilege-change	Allows the user to change the set of privileges assigned to a user, or to change the set of default root privileges.
server-restart	Allows the user to request the server restart task.
server-shutdown	Allows the user to request the server shutdown task.
subentry-write	Allows the associated user to perform LDAP subentry write operations.
unindexed-search	Allows the user to request unindexed search operations.
update-schema	Allows the user to update the server schema.

The following table lists the privileges that can be assigned to the root user.

Privilege	Description
jmx-notify	Allows the user to subscribe to JMX notifications.
jmx-read	Allows the user to read JMX attribute values.
jmx-write	Allows the user to update JMX attribute values.
proxied-auth	Allows the user to use the proxied authorization control or to request an alternate SASL authorization ID.

29.3 Managing Root Users With dsconfig

Use the dsconfig command to manage root users.

For more information, see Managing the Server Configuration Using dsconfig.

The following topics list the tasks to manage root users using dsconfig command:

- Viewing the Default Root User Privileges
- Editing the Default Root User Privileges
- Creating a Root User
- Changing a Root User's Password
- Changing a Root User's Privileges

29.3.1 Viewing the Default Root User Privileges

The default root user has various privileges, which are stored as values of the default-root-privilege-name property.

View the default root user privileges by running the following dsconfig command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-root-dn-prop

Property                    : Value(s)
----------------------------:-------------------------------------------------
default-root-privilege-name : backend-backup, backend-restore, bypass-acl,
                            : cancel-request, config-read, config-write,
                            : disconnect-client, ldif-export, ldif-import,
                            : modify-acl, password-reset, privilege-change,
```

```
                                        : server-restart, server-shutdown,
                                        : unindexed-search, update-schema
```

29.3.2 Editing the Default Root User Privileges

You can add or remove privileges for the default root user or add or remove the values of the `default-root-privilege-name` property. The easiest way to manage root user privileges is to use `dsconfig` in interactive mode. Interactive mode walks you through the root user configuration, and is therefore not documented here.

The `default-root-privilege-name` property holds the following values:

- `backend-backup`
- `backend-restore`
- `bypass-acl`
- `cancel-request`
- `config-read`
- `config-write`
- `disconnect-client`
- `jmx-notify`
- `jmx-read`
- `jmx-write`
- `ldif-export`
- `ldif-import`
- `modify-acl`
- `password-reset`
- `privilege-change`
- `proxied-auth`
- `server-restart`
- `server-shutdown`
- `unindexed-search`
- `update-schema`

The following example adds the `jmx-notify` privilege to the default root user, by using `dsconfig` in non-interactive mode.

Run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-root-dn-prop --add default-root-privilege-name:jmx-notify
```

29.3.3 Creating a Root User

Use the `ldapmodify` command to create a new root user and to create the entry in LDIF. Root users are stored below the entry `cn=Root DNs,cn=config`.

> **Note:**
>
> The `cn=config` suffix is available only through the administration connector, and must therefore be accessed over SSL, through the administration port.

Root users automatically inherit the set of default root user privileges on the server.

To create a root user:

1. Create the root user entry below the `cn=Root DNs,cn=config` entry.

   The following LDIF file represents a new root user named "Administration Manager". The entry is saved in a file named `add-root-user.ldif`.

   ```
   dn: cn=MyRootUser,cn=Root DNs,cn=config
   objectClass: inetOrgPerson
   objectClass: person
   objectClass: top
   objectClass: ds-cfg-root-dn-user
   objectClass: organizationalPerson
   userPassword: password
   cn: MyRootUser
   sn: MyRootUser
   ds-cfg-alternate-bind-dn: cn=MyRootUser
   givenName: Directory
   ```

2. Use the `ldapmodify` command to add the entry.

   ```
   $ ldapmodify -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file \
     --useSSL --defaultAdd --filename "add-root-user.ldif"
   Processing ADD request for cn=MyRootUser,cn=Root DNs,cn=config
   ADD operation successful for DN cn=MyRootUser,cn=Root DNs,cn=config
   ```

3. Use the `ldapsearch` command to display all the root users defined in the server.

   ```
   $ ldapsearch -p 4444 -b "cn=root DNs,cn=config" -D "cn=directory manager" -j pwd-
   file \
   --useSSL "objectclass=*" dn
   dn: cn=Root DNs,cn=config

   dn: cn=MyRootUser,cn=Root DNs,cn=config

   dn: cn=Directory Manager,cn=Root DNs,cn=config
   ```

For information about adding or removing privileges for a specific root user, see Changing a Root User's Privileges.

29.3.4 Changing a Root User's Password

Use the `ldappasswordmodify` client command to change the password of a root user.

To change a root user's password:

1. Create a password in a secure file.
2. Use `ldappasswordmodify` to change the password.

```
$ ldappasswordmodify -h localhost -p 4444 -D "cn=MyRootUser" -j pwd-file \
  --useSSL --newPasswordFile rootuser_pwd.txt
The LDAP password modify operation was successful
```

29.3.5 Changing a Root User's Privileges

If you want to have a different set of privileges for a specific root user, add the `ds-privilege-name` attribute to that root user's entry.

The following example gives the root user `"cn=MyRootUser,cn=Root DNs,cn=config"` the ability to use proxied authorization. The example removes the ability to change user privileges or access the configuration. The minus sign before the privilege indicates that the privilege is being removed rather than granted.

Apply the following LDIF statement to the root user's entry:

```
dn: cn=MyRootUser,cn=Root DNs,cn=config
changetype: modify
add: ds-privilege-name
ds-privilege-name: proxied-auth
ds-privilege-name: -config-read
ds-privilege-name: -config-write
```

In this example, the root user `"cn=MyRootUser,cn=Root DNs,cn=config"` would inherit all privileges automatically granted to root users with the exception of the `config-read` and `config-write` privileges. The user would also be given the `proxied-auth` privilege.

29.4 Setting Root User Resource Limits

You can set resource limits on the server for search operations by using the operational attributes on the client application that is binding to the server.

The following resource limits are available:

- **Look-through limit**. Specify the maximum number of entries that can be examined during a single search operation. Use the `ds-rlim-lookthrough-limit` operational attribute.
- **Size limit**. Specify the maximum number of entries that can be returned in a single search operation. Use the `ds-rlim-size-limit` operational attribute.
- **Time limit**. Specify the maximum length of time in seconds that the server can spend processing a search operation. Use the `ds-rlim-time-limit` operational attribute.

The following LDIF update statement sets resource limits for the new root user created in the previous section. This statement should be applied to the root user's entry.

```
dn: cn=MyRootUser,cn=Root DNs,cn=config
changetype: modify
add: ds-rlim-lookthrough-limit
ds-rlim-lookthrough-limit: 1000
-
add: ds-rlim-size-limit
ds-rlim-size-limit: 500
-
add: ds-rlim-time-limit
ds-rlim-time-limit: 300
```

To set a particular resource limit to *unlimited*, set the value of the corresponding attribute to `0` (zero).

29.5 Managing Administrators

An administrator generally has broader rights and permissions than most users. By default, administrators are not replicated because they are stored in the OUD configuration. You can create several administrators, each with different access controls and resource limits.

When you set up replication servers using the graphical installer or the `dsreplication` command, you are prompted to set a user name and password for the Global Administrator. The Global Administrator is responsible for managing and maintaining administrative server domains in replicated environments.

> **Note:**
>
> Only root users can bind to the administration port because administrative binds are resolved with root `dns` from `cn=config`.

More information on administrators is explained in the following sections:

- Viewing the Global Administrator Entry
- Creating Administrators with Limited Privileges

29.5.1 Viewing the Global Administrator Entry

The Global Administrator created for the replication exists in the `cn=Administrators,cn=admin data` subtree, so it is replicated and can be used with every OUD instance of a replicated topology.

To view the Global Administrator entry, run the following `ldapsearch` command:

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file \
  --useSSL -b "cn=Administrators,cn=admin data" -s sub "(objectclass=*)"
dn: cn=Administrators,cn=admin data
objectClass: top
objectClass: groupofurls
description: Group of identities which have full access.
cn: Administrators
memberURL: ldap:///cn=Administrators,cn=admin data??one?(objectclass=*)
dn: cn=admin,cn=Administrators,cn=admin data
objectClass: person
objectClass: top
userPassword: {SSHA}+ed1wbhcWjxtv2zJ6OHEA2TuE9n1qIJGnuR94w==
description: The Administrator that can manage all the OUD instances.
cn: admin
```

29.5.2 Creating Administrators with Limited Privileges

The Global Administrator created for the replication has the full set of administrator privileges. In some situations, it might be useful to create additional administrators having only a subset of administrator rights. For example, a Monitor Administrator would have the privilege to read the OUD configuration, but would not be able to modify it. Custom administrators are stored in a

replicated suffix `cn=admin data`. Like Global Administrators, custom administrators are replicated.

To create an administrator with limited privileges, create your own administrator container node in the `cn=admin data` suffix:

```
./ldapmodify -a -p 4444 -Z -X -D "cn=directory manager"  -w ****
dn: cn= my admins,cn=admin data
objectclass: top
objectClass: ds-cfg-branch

dn: cn=monitor,cn=my admins,cn=admin data
objectClass: person
cn: monitor
sn: monitor
userpassword: ****
```

At this stage, it is possible to use these credentials (`cn=monitor,cn=my admins,cn=admin data`) with `dsconfig`. The `dsconfig` command can authenticate that user, however the administrator won't be able to read the configuration because the administration does not have the privilege to do so. The `dsconfig` command reports the following error during navigation in the configuration:

```
The Administration Connector could not be modified because you do not
have the correct authorization
```

You must assign the appropriate privileges giving the administrator the right to perform the desired actions. In the previous example, the administrator must be assigned the `config-read` privilege. The `bypass-acl` privilege is also required so that the administrator can perform privileged actions on the configuration.

```
./ldapmodify -p 4444 -Z -X -D "cn=directory manager"  -w ****
dn: cn=monitor,cn=my admins,cn=admin data
changetype: modify
add: ds-privilege-name
ds-privilege-name: bypass-acl
ds-privilege-name: config-read
```

Now the administrator can read the configuration using `dsconfig`. However, any attempt to modify the configuration would result in the following error:

```
The Configuration could not be modified because you do not have
the correct authorization.
```

30
Managing Password Policies

A *password policy* is a set of rules governing the use of passwords in the system and it is an integral component of any security strategy employed for your directory. Oracle Unified Directory includes a default password policy for general users and a default password policy for root users. These default password policies reside in the directory server's configuration and they can be modified.

In addition to default password policies, Oracle Unified Directory supports multiple password policies, which allows you to create and configure specialized password policies for a specific set of users. Customized password policies can be defined as LDAP subentries and stored with the user data, which allows the policies to be replicated across servers.

Oracle Unified Directory uses the `dsconfig` utility and Oracle Unified Directory Services Manager (OUDSM) to configure and manage password policies.

Topics:

30.1 Understanding Password Policy Components

Review this topic for the various components that are configurable in all password policies.

All password policies involve the following configurable components:

- **Password complexity requirements**. Specifies the password's composition and required number of characters. Typically, you would specify the minimum number of characters used in a password, the type of characters allowed, and the required number of numeric characters. For example, many institutions require a minimum of seven or eight characters, one numeral, one special character, as well as a mix of uppercase and lowercase letters.
- **Password history**. Determines the number of unique passwords that users must use before they can reuse an old password.
- **Maximum password age**. Determines how long users can use a password before they are allowed, or required, to change it.
- **Minimum password age**. Determines how long users must keep a new password before they can change it.

- **First Login**. Determines if users are required to change their password when they first log in to the system.
- **Authorized password change**. Refers to the conditions under which users can change their password. For example, you can configure the server so that before users can change their password, they must enter their current password to authenticate their identity before entering a new password.
- **Account lockout**. Determines under which conditions an account is disabled for access by the user. For example, you can configure the server to that if a user fails to properly authenticate after three attempts, then the account will be locked on the fourth attempt. After which, an administrator must manually unlock the account for that user.
- **Password storage scheme**. Determines how to encrypt the password and store it on the server. You can configure storage schemes for certain accounts on the server. For example, root user passwords require strong encryption due to the importance of the account and its privileges. Thus, you can configure the use the SSHA-512 storage scheme to store root user passwords.

> **Note:**
>
> Oracle Unified Directory provides a `Password Expiration Time` virtual attribute that can dynamically compute the exact time when a user's password will expire, based on information contained in both the user entry and the applicable password policy.
>
> For more information about virtual attributes, see Configuring Virtual Attributes.

Password validation is not handled directly in the password policy, but by specific password validator entries, the DNs of which are present in the password policy. For more information, see Managing Password Validators.

30.2 Working with the Default Password Policy Properties

Review these topics for a list of the default password policy properties and how to manage these properties.

- Default Password Policy Properties
- Viewing the Properties of the Default Password Policy
- Modifying the Default Password Policy

30.2.1 Default Password Policy Properties

Review this topic for a list of all the properties in a default password policy and their descriptions.

The following table lists the default password policy properties:

Table 30-1 Default Password Policy Properties

Property	Description
`account-status-notification-handler`	Sends messages when events occur during password policy processing. Use this property to specify the DNs of the account status notification handlers to use for this password policy.

Table 30-1 (Cont.) Default Password Policy Properties

Property	Description
`allow-expired-password-changes`	*Not recommended*. Indicates whether users are allowed to change their passwords after the passwords have expired. The user must issue the request anonymously and include the current password in the request. If enabled, this feature uses the Password Modify Extended Operation, which is enabled by default at initial configuration.
`allow-user-password-changes`	Indicates whether users are allowed to change their own passwords if they have access control rights to do so.
`default-password-storage-scheme`	Specifies the password storage scheme that is used to encode clear-text passwords for this password policy. See password storage scheme.
`deprecated-password-storage-scheme`	Specifies the DNs for password storage schemes that are considered deprecated for this password policy. If a user with this password policy authenticates to the server and his password is encoded with any deprecated schemes, those values are removed and replaced with values encoded using the default password storage scheme.
`expire-password-without-warning`	Indicates whether user passwords are allowed to expire even if the user has not yet seen a password expiration warning. If this is set to `false`, the user is always guaranteed to see at least one warning message even if the password expiration time has passed. The expiration time will be reset to the current time plus the warning interval (`ds-cfg-password-expiration-warning-interval`).
`force-change-on-add`	Indicates whether users are required to change their passwords the first time they use their accounts and before they are allowed to perform any other operation.
`force-change-on-reset`	Indicates whether users are required to change their passwords after an administrative password reset and before they are allowed to perform any other operation.
`grace-login-count`	Specifies the maximum number of grace login that a user should be given. A grace login makes it possible for a user to authenticate to the server even after the password has expired, but the user is not allowed to do anything else until he has changed his password.
`idle-lockout-interval`	Specifies the maximum length of time that a user account can remain idle (that is, that the user may go without authenticating to the directory) before the server locks the account. This action is enforced if `last login time` tracking is enabled and if the `idle lockout interval` is set to a nonzero value.
`last-login-time-attribute`	Specifies the name of the attribute in the user's entry that is used to hold the last login time for the user. If this is provided, the specified attribute must either be defined as an operational attribute in the server schema, or it must be allowed by at least one of the object classes in the user's entry. The `ds-pwp-last-login` operational attribute has been defined for this purpose. Last login time tracking is only enabled if the `ds-cfg-last-login-time-attribute` and `ds-cfg-last-login-time-format` attributes have been configured for the password policy.

Table 30-1 (Cont.) Default Password Policy Properties

Property	Description
last-login-time-format	Specifies the format string that should be used to generate the last login time values, which can be any valid format string that can be used with the java.text.SimpleDateFormat class. **Note**: For performance reasons, it might be desirable to configure this attribute so that it only stores the date (format: yyyyMMdd) and not the time of the last login. Then, it must only be updated once per day, rather than each time the user may authenticate. Last login time tracking is only enabled if the ds-cfg-last-login-time-attribute and ds-cfg-last-login-time-format attributes have been configured for the password policy.
last-login-time-zone	Specifies the Time Zone String that should be used to generate the last login time value, which can be any valid time zone string. Based on the last-login-time-zone that you specify, the last-login-time is generated in the same time zone. For example, if you set this attribute to EST, the last-login-time value is generated in EST.
lockout-duration	Specifies the length of time that a user account should remain locked due to failed authentication attempts before it is automatically unlocked. A value of "0 seconds" indicates that any locked accounts are not automatically unlocked and must be reset by an administrator.
lockout-failure-count	Specifies the number of authentication failures required to lock a user account, either temporarily or permanently. A value of zero indicates that automatic lockout is not enabled.
lockout-failure-expiration-interval	Specifies the maximum length of time that a previously failed authentication attempt should be counted toward a lockout failure. **Note**: The record of all previous failed attempts is always cleared upon a successful authentication. A value of "0 seconds" indicates that failed attempts are never automatically expired.
lockout-soft-duration-count	Specifies the length of time that an account is temporarily locked after too many authentication failures. The value of this attribute is an integer followed by a unit of seconds, minutes, hours, days, or weeks. A value of 0 seconds indicates that the account must never be locked temporarily.
lockout-soft-failure-count	Specifies the maximum number of authentication failures that a user is allowed before the account is locked temporarily. A value of 0 indicates that accounts are never locked temporarily due to failed attempts.
max-password-age	Specifies the maximum length of time that a user is allowed to keep the same password before choosing a new one. This is often known as the *password expiration interval*. A value of "0 seconds" indicates that passwords never expire. If the ds-cfg-expire-passwords-without-warning attribute is set to false, the effective password expiration time is recalculated to be the time at which the first warning is received, plus the warning interval (ds-cfg-password-expiration-warning-interval). This behavior ensures that a user always has the full configured warning interval to change his password.

Table 30-1 (Cont.) Default Password Policy Properties

Property	Description
max-password-reset-age	Specifies the maximum length of time that users are allowed to change their passwords after they have been administratively reset and before they are locked out. This is only applicable if the ds-cfg-force-change-on-reset attribute is set to true. A value of "0 seconds" indicates that there are no limits on the length of time that users have to change their passwords after administrative resets.
min-password-age	Specifies the minimum length of time that a user is required to have a password value before it can be changed again. Providing a nonzero value ensures that users are not allowed to repeatedly change their passwords to flush their previous password from the history so it can be reused.
password-attribute	Specifies the attribute in the user's entry that holds the encoded passwords for the user. The specified attribute must be defined in the server schema, and it must have either the user password syntax or the authentication password syntax. Typically, you enter "userPassword" for the User Password syntax (OID: 1.3.6.1.4.1.26027.1.3.1). You can also specify, if your server supports it, the value authPassword for the authenticated password syntax (OID: 1.3.6.1.4.1.4203.1.1.2).
password-change-requires-current-password	Indicates whether users are required to provide their current password when setting a new password. If this is set to true, then users are required to provide their current password when changing their existing password. This may be done using the password modify extended operation, or using a standard LDAP modify operation by deleting the existing password value and adding the new password value in the same modify operation.
password-expiration-warning-interval	Specifies the length of time before the password expires that the users should start to receive notification that it is about to expire. This must be given a nonzero value if the ds-cfg-expire-passwords-without-warning attribute is set to false.
password-generator	Specifies the DN for the password generator that should be used with this password policy. The password generator is used with the password modify extended operation to provide a new password for cases in which the client did not include one in the request. If no password generator DN is specified, then the password modify extended operation does not automatically generate passwords for users.

Table 30-1 (Cont.) Default Password Policy Properties

Property	Description
password-history-count	Specifies the maximum number of password values that should be maintained in the password history. Whenever a user's password is changed, the server checks the proposed new password against the current password and all passwords stored in the history. If a match is found, then the user is not allowed to use that new password. A value of zero indicates either that the server should not maintain a password history (that is, the password history duration has a value of "0 seconds") or that the password history list should be based entirely on duration and no maximum count should be enforced (that is, the password history duration has a value other than "0 seconds"). **Note**: If an administrator reduces the configured password history count to a smaller (but still nonzero) value, each user entry containing password history state information is not impacted until a password change is processed for that user. At that time, any excess history state values is purged from the entry. If the history count is reduced to zero and the password history duration is also set to "0 seconds," any state information in the user's entry is retained in case the feature is reenabled.
password-history-duration	Specifies the maximum length of time that a formerly used password should remain in effect in the user's password history. Whenever a user's password is changed, the server checks the proposed new password against the current password and all passwords stored in the history. If a match is found, the user is not allowed to use that new password. A value of "0 seconds" indicates either that the server should not maintain a password history (that is, the password history count has a value of "0") or that the password history list should be based entirely on count and no maximum duration should be enforced (that is, the password history count has a value other than "0").
password-validator	Specifies the DNs for password validators that should be used with this password policy. The password validators are invoked whenever a user attempts to provide a new password to determine whether that new password is acceptable.
previous-last-login-time	Indicates the next-to-last time that the user authenticated to the server using a BIND operation. When the user logs in, Oracle Unified Directory copies the existing last-login-time value (in the format that was used when it was written, and only at that time) to previous-last-login-time, and then updates the last-login-time value to reflect the newer login time.
previous-last-login-time-format	Specifies the format string that was used in the past for older last login time values. This value is not necessary unless the last-login-time option is enabled and the format in which the values are stored has been changed.
require-change-by-time	Specifies a time by which all users with this password policy are required to change their passwords. This option works independently of password expiration (that is, force all users to change their passwords at some point even if password expiration is disabled).
require-secure-authentication	Indicates whether users with this password policy are required to authenticate in a secure manner using a secure communication mechanism like SSL, or a secure SASL mechanism like DIGEST-MD5, EXTERNAL, or GSSAPI that does not expose the password in the clear.

Table 30-1 (Cont.) Default Password Policy Properties

Property	Description
`require-secure-password-changes`	Indicates whether users with this password policy are required to make password changes in a secure manner, such as over a secure communication channel like SSL.

30.2.2 Viewing the Properties of the Default Password Policy

You can either use Oracle Unified Directory Services Manager or the `dsconfig` command to display the properties of the default password policy.

- Viewing Default Password Policy Properties Using `dsconfig`
- Viewing Default Password Policy Properties Using OUDSM

30.2.2.1 Viewing Default Password Policy Properties Using `dsconfig`

To view the properties using `dsconfig`, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-password-policy-prop --policy-name "Default Password Policy"

Property                                  : Value(s)
------------------------------------------:--------------------------
account-status-notification-handler       : -
allow-expired-password-changes            : false
allow-user-password-changes               : true
default-password-storage-scheme           : Salted SHA-1
deprecated-password-storage-scheme        : -
expire-passwords-without-warning          : false
force-change-on-add                       : false
force-change-on-reset                     : false
grace-login-count                         : 0
idle-lockout-interval                     : 0 s
last-login-time-attribute                 : -
last-login-time-format                    : -
last-login-time-zone                      : UTC
lockout-duration                          : 0 s
lockout-failure-count                     : 0
lockout-failure-expiration-interval       : 0 s
max-password-age                          : 0 s
max-password-reset-age                    : 0 s
min-password-age                          : 0 s
password-attribute                        : userpassword
password-change-requires-current-password : false
password-expiration-warning-interval      : 5 d
password-generator                        : Random Password Generator
password-history-count                    : 0
password-history-duration                 : 0 s
password-validator                        : -
previous-last-login-time-format           : -
require-change-by-time                    : -
require-secure-authentication             : false
require-secure-password-changes           : false
```

To view any advanced properties, include the `--advanced` option, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-password-policy-prop --policy-name "Default Password Policy" --advanced
```

30.2.2.2 Viewing Default Password Policy Properties Using OUDSM

To view the properties using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.
4. Select **Default Password Policy**.

The password policy properties, and their values, are displayed in the right-hand pane.

30.2.3 Modifying the Default Password Policy

You can either use Oracle Unified Directory Services Manager or the `dsconfig` command to modify the different properties of the default password policy.

- Modifying Default Password Policy Properties Using `dsconfig`
- Modifying Default Password Policy Properties Using OUDSM

30.2.3.1 Modifying Default Password Policy Properties Using `dsconfig`

To modify the properties by using `dsconfig`, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop --policy-name "Default Password Policy" \
  --set allow-expired-password-changes:true
```

30.2.3.2 Modifying Default Password Policy Properties Using OUDSM

To modify the properties by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.
4. Select **Default Password Policy**.

   The password policy properties, and their values, are displayed in the right-hand pane.
5. Modify the required property and click **Apply**.

You cannot display or modify advanced properties by using OUDSM.

30.3 Attributes for Password Policy State Information

Password policy state information must be maintained for each user. This information is stored in each user entry as a set of operational attributes, which are typically declared with the `NO-`

USER-MODIFICATION flag to prevent them from being directly modified by end users or administrators.

The password policy includes many operational attributes to maintain the state information, as described in the following table.

Table 30-2 Password Policy Operational Attributes

Attribute	Description
pwdChangedTime	This attribute holds the time stamp (in generalized time format) of the last time that the user's password was changed, either by that user or by an administrator. It is automatically set on an add, modify, or password modify operation that sets or alters the user's password, and it should never be cleared or unset. It will be used to determine when the user's password was last changed for the purposes of enforcing the minimum and maximum password ages, and to determine whether to generate expiration warning notifications. It will also be used with the pwdReset attribute to enforce the maximum password reset age.
pwdGraceUseTime	This attribute holds the time stamps (in generalized time format) of the times that a user authenticated with a grace login after that user's password had expired, to ensure that the maximum number of grace login is enforced. This is automatically set whenever the user authenticates using one of the grace logins, and it is cleared whenever the user's password is changed by that user or reset by an administrator.
pwdFailureTime	This attribute holds the time stamps (in generalized time format) of the times that an authentication attempt failed for the user because the wrong password was provided. It is used to enforce the maximum failure account, so that an account may be locked as a result of too many failed attempts. This is set automatically whenever such an authentication failure occurs, and is cleared whenever the user authenticates successfully (whether before the lockout occurs or after the account has been locked and the lockout duration has passed) or whenever the user's password is changed by that user or reset by an administrator.
pwdHistory	This attribute holds previous passwords with a time stamp (in generalized time format). It is used if you have set ds-cfg-password-history-duration, ds-cfg-password-history-count, or both. This is set automatically when you change passwords.

Table 30-2 (Cont.) Password Policy Operational Attributes

Attribute	Description
pwdAccountLockedTime	This attribute holds the time stamp (in generalized time form) of the time that the user's account was locked after too many failed authentication attempts. It is used to indicate that the account is locked, and to provide information about when the account may be automatically unlocked through the password lockout duration. It is automatically cleared if the user's password is reset by an administrator, or on any authentication attempt (regardless of its success or failure) after the lockout duration has passed. **Note:** The Oracle Unified Directory password policy implementation does vary from the behavior specified in the password policy draft in one significant way. In the Oracle Unified Directory implementation, this attribute will always hold the time that the account was locked, regardless of whether the account lockout is temporary or permanent. The password policy draft states that in the event that the account should not be automatically unlocked after some period of time, it should be given a special value of 00000101000000Z. There are several justifications for this variation, but the primary reasons are that the time specified in the draft is actually illegal (the Gregorian calendar does not have a year 0), and this special value is unnecessary because the determination about whether the account is locked temporarily or permanently may be made based on the value of the ds-cfg-lockout-duration attribute (a value of 0 seconds indicates that the account should not be automatically unlocked).
pwdPolicySubEntry	This attribute holds the password policy for a given entry. Each object that is controlled by password policy advertises the subentry that is being used to control its policy in its pwdPolicySubentry attribute. Users wishing to examine or manage password policy for an object may interrogate the pwdPolicySubentry for that object to arrive at the proper pwdPolicy subentry.
ds-pwp-password-policy-dn	This attribute holds the DN of the configuration entry for the password policy that should be enforced for the associated user. If it is defined, then it must refer to a valid existing password policy definition configuration entry or subentry. If this attribute exists in a user's entry, but does not refer to a valid configuration entry or subentry, then the user is not allowed to authenticate. You can use the pwdPolicySubentry operational attribute to verify which policy is in effect for each specific user entry.
pwdReset	This attribute holds a Boolean value of true if the user's password has been reset by an administrator and must be changed before the user is allowed to perform any other kind of operation. It will be automatically set to true when the user's account is added if the ds-cfg-force-change-on-add attribute is set to true, or on an administrative modify or password modify operation that resets the user's password if the ds-cfg-force-change-on-reset attribute is set to true. It is automatically cleared whenever the user's password is changed by that user.
ds-pwp-account-disabled	This attribute holds a Boolean value of true if the user's account has been manually disabled by an administrator, in which case that user is not allowed to authenticate to the directory server. This attribute is never automatically set or cleared by the directory server, but must be manually specified by the administrator, or may be generated as a virtual attribute.

Table 30-2 (Cont.) Password Policy Operational Attributes

Attribute	Description
ds-pwp-last-login-time	This attribute is provided for use as the default attribute for holding last login time information if that feature should be enabled. If that feature is enabled, then there is no requirement that this attribute be used, and an alternate attribute may be configured if the administrator so chooses.
ds-pwp-password-changed-by-required-time	This attribute may hold a generalized time value that is equal to the value of the ds-cfg-require-change-by-time attribute in the password policy configuration entry. It is used to indicate whether the user's password has been changed in accordance with that configuration. This attribute is automatically set to the value of the ds-cfg-require-change-by-time attribute whenever the user's password is changed (by the end user or an administrator) any time that configuration attribute has a value that is different from the value currently held in the ds-pwp-password-changed-by-required-time attribute.
ds-pwp-warned-time	This attribute holds a time stamp (in generalized time form) that indicates when the user was first warned about an upcoming password expiration. It is used with the ds-cfg-expire-passwords-without-warning configuration attribute to determine whether a user has seen an expiration warning and if so what the new adjusted expiration time should be. It is automatically set by the directory server the first time that a warning notification is sent to indicate that a password is about to expire, and it is cleared whenever the user's password is changed (either by that user or an administrator).

30.4 Attributes Used in the pwdPolicy Object Class

The pwdPolicy object class contains the attributes that define a password policy in effect for a set of users.

The following schema definition for the pwdPolicy object class depicts the attributes supported by the LDAP subentry pwdPolicy:

```
( 1.3.6.1.4.1.42.2.27.8.2.1
        NAME 'pwdPolicy'
        SUP top
        AUXILIARY
        MUST ( pwdAttribute )
        MAY ( pwdMinAge $ pwdMaxAge $ pwdInHistory $ pwdCheckQuality $
        pwdMinLength $ pwdExpireWarning $
        pwdGraceAuthNLimit $ pwdLockout $
        pwdLockoutDuration $ pwdMaxFailure $ pwdFailureCountInterval $
        pwdMustChange $ pwdAllowUserChange $ pwdSafeModify ) )
```

Table 30-3 describes the attributes supported by the pwdPolicy objectclass.

Table 30-3 Attributes Supported by the pwdPolicy ObjectClass

Attribute	Description
pwdAttribute	This holds the name of the attribute to which the password policy is applied. For example, the password policy may be applied to the userPassword attribute.
pwdMinAge	This attribute holds the number of seconds that must elapse between modifications to the password. If this attribute is not present, 0 seconds is assumed.
pwdMaxAge	This attribute holds the number of seconds after which a modified password will expire. If this attribute is not present, or if the value is 0 the password does not expire. If not 0, then the value must be greater than or equal to the value of the pwdMinAge.
pwdInHistory	This attribute specifies the maximum number of used passwords stored in the pwdHistory attribute. If this attribute is not present, or if the value is 0, then the used passwords are not stored in the pwdHistory attribute and thus may be reused.
pwdCheckQuality	This attribute indicates how the password quality will be verified while being modified or added. If this attribute is not present, or if the value is 0, then quality checking is not enforced. A value of 1 indicates that the server will check the quality, and if the server cannot check it (due to a hashed password or other reasons) it will be accepted. A value of 2 indicates that the server will check the quality, and if the server cannot verify it, it will return an error refusing the password.
pwdMinLength	When quality checking is enabled, this attribute holds the minimum number of characters that must be used in a password. If this attribute is not present, no minimum password length will be enforced. If the server cannot check the length (due to a hashed password or otherwise), the server will, depending on the value of the pwdCheckQuality attribute, either accept the password without checking it (0 or 1) or refuse it (2).
pwdExpireWarning	This attribute specifies the maximum number of seconds before a password is due to expire that expiration warning messages will be returned to an authenticating user. If this attribute is not present, or if the value is 0 no warnings will be returned. If not 0, then the value must be smaller than the value of the pwdMaxAge attribute.
pwdGraceAuthNLimit	This attribute specifies the number of times an expired password can be used to authenticate. If this attribute is not present or if the value is 0, authentication will fail.
pwdLockout	This attribute indicates, when its value is TRUE, that the password may not be used to authenticate after a specified number of consecutive failed bind attempts. The maximum number of consecutive failed bind attempts is specified in pwdMaxFailure attribute. If this attribute is not present, or if the value is FALSE, the password may be used to authenticate when the number of failed bind attempts has been reached.

Table 30-3 (Cont.) Attributes Supported by the pwdPolicy ObjectClass

Attribute	Description
pwdLockoutDuration	This attribute holds the number of seconds that the password cannot be used to authenticate due to too many failed bind attempts. If this attribute is not present, or if the value is 0 the password cannot be used to authenticate until reset by a password administrator.
pwdMaxFailure	This attribute specifies the number of consecutive failed bind attempts after which the password may not be used to authenticate. If this attribute is not present, or if the value is 0, this policy is not checked, and the value of pwdLockout will be ignored.
pwdFailureCountInterval	This attribute holds the number of seconds after which the password failures are purged from the failure counter, even though no successful authentication occurred. If this attribute is not present, or if its value is 0, the failure counter is only reset by a successful authentication.
pwdMustChange	This attribute specifies with a value of TRUE that users must change their passwords when they first bind to the directory after a password is reset by a password administrator. If this attribute is not present, or if the value is FALSE, users are not required to change their password upon binding after the password administrator resets the password. This attribute is not set due to any actions specified by this document, it is typically set by a password administrator after resetting a user's password.
pwdAllowUserChange	This attribute indicates whether users can change their own passwords, although the change operation is still subject to access control. If this attribute is not present, a value of TRUE is assumed. This attribute is intended to be used in the absence of an access control mechanism.
pwdSafeModify	This attribute specifies whether the existing password must be sent along with the new password when being changed. If this attribute is not present, a FALSE value is assumed.

30.5 Understanding Password Policies, Password Validators, and Password Generators in a Replicated Environment

You can understand about the policies governing password in a replicated environment. The password policies, password validators, or password generators that reside in the directory server configuration (under cn=config) are not replicated. Configuration information in general is not replicated and is specific to each directory server instance.

If you modify the default password policies, password validators, or password generators, you must make the same changes on each directory server instance in a replicated topology. Similarly, specialized password policies, password validators, or password generators under cn=config are not replicated to other directory servers.

Password policies/Password Validators/Password Generators that are created as subentries (that is, as part of the data) are replicated.

For information about creating password policies as subentries, see Defining a Password Policy as an LDAP Subentry

For information about creating password validators as subentries, see Defining a Password Validator as an LDAP Subentry

For information about creating password generators as subentries, see Defining a Password Generator as an LDAP Subentry

Additional considerations for using password policies in replicated environments include the following:

- The directory server replicates all password information (current password, password history, password expiration) that is stored in the user entry.
- If a user changes his password, the new password might take a while to be updated on all replicas.
- A user might receive multiple password expiration warnings, one from each replicated server.

30.6 Managing Password Policies by Using the Command Line

The easiest way to configure a password policy is by using the command line. Use the `dsconfig` command to manage the existing password policies and to modify the password policy properties.

This section contains the following topics:

- Configuring the Default Password Policy
- Creating a New Password Policy
- Creating a First Login Password Policy
- Assigning a Password Policy to an Individual Account
- Preventing Password Policy Modifications
- Assigning a Password Policy to a Group of Users
- Defining a Password Policy as an LDAP Subentry
- Deleting a Password Policy

30.6.1 Configuring the Default Password Policy

Use the `dsconfig` command to modify various properties of the default password policy.

- Account Lockout Features
- Configuring Last Login
- Configuring Password History Count and Duration

30.6.1.1 Account Lockout Features

The following table lists the account lockout features:

Table 30-4 Account lockout features

Features	Description
Lockout failure count.	he `lockout-failure-count` property specifies the number of authentication failures required to lock a user account

Table 30-4 (Cont.) Account lockout features

Features	Description
Lockout soft failure count.	The `lockout-soft-failure-count` property specifies the number of authentication failures required to soft lock a user account
Lockout duration.	The `lockout-duration` property determines the length of time that the account is in a locked state after failed authentication attempts. A value of zero indicates that the account is not automatically unlocked
Soft Lockout duration	The `lockout-soft-duration` property determines the length of time that the account is in a soft-locked state after failed authentication attempts. After the soft lockout duration expires, the account is automatically unlocked.
Lockout failure expiration interval.	The `lockout-failure-expiration-interval` property determines the maximum length of time that a previously failed authentication attempt should be counted toward a lockout failure. A value of zero indicates that failed attempts never automatically expire
Idle lockout interval.	The `idle-lockout-interval` property specifies the maximum length of time that a user account can go without authenticating to the directory before the server locks the account. This property is enforced if the `last-login-time` is enabled and `idle-lockout-interval` is set to a nonzero value.

The following command sets the account lockout properties for the default password policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop \
  --policy-name "Default Password Policy" --set "lockout-soft-failure-count:3" \
  --set "lockout-duration:15 minutes" --set "idle-lockout-interval:90 days" \
  --set "lockout-failure-expiration-interval:10 minutes"
```

The following command sets the account lockout properties for a password policy using a hard account lock.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop \
  --policy-name "Default Password Policy" --set "lockout-failure-count:9"\
  --set "lockout-soft-failure-count:3" --set "lockout-duration:0 seconds"\
  --set "lockout-soft-duration:10 minutes"\
  --set "idle-lockout-interval:90 days"\
  --set "lockout-failure-expiration-interval:10 minutes"
```

In this example, if the user fails to log in twice, the system times out on the third failed attempt.

After the `lockout-soft-duration` period expires, the user again fails three attempts to log in. The user account is locked for the `lockout-soft-duration` of 10 minutes.

After the `lockout-failure-expiration-interval` of 10 minutes elapses, an authentication failure is no longer counted against a user for the purposes of account lockout. This helps to prevent unauthorized people from trying to guess your password using multiple login attempts over a short period of time.

After the second `lockout-soft-duration` period expires, the user again fails three attempts to log in. The user account is now hard locked, and the account must be manually unlocked by an administrator.

30.6.1.2 Configuring Last Login

Last login is a basic security feature that helps the user to keep track of the login history. The directory server provides an operational attribute, `ds-pwp-last-login`, that holds the user's last login time. If you specify another attribute, the operational attribute must be defined in the server schema, or it must be allowed by at least one of the object classes in the user's entry.

The `last-login-time-format` property determines the time format, for example *yyyMMdd* or 20140922. If the time format has changed, and `last-login` is enabled, the `previous-last-login-time-format` property might be used to decode a user's login time, if the latter does not match the `last-login-time-format` syntax.

The `last-login-time-zone` property determines the time zone, for example *EST*. Based on the `last-login-time-zone` that you specify, the `last-login-time` is generated in the same time zone. For example, if the `last-login-time-zone` property is set to `EST`, the `last-login-time` value is generated in `EST`.

The `previous-last-login-time` property attribute holds the user's *next-to-last* login time. Oracle Unified Directory obtains this value from the `last-login-time` value, and displays the `previous-last-login-time` value in whatever format was used when it was written, and only at that time. When a new login occurs, Oracle Unified Directory copies the existing `last-login-time` value to `previous-last-login-time`, and updates the `last-login-time` value to reflect the newer login time.

The following command sets the last login properties for the default password policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop \
  --policy-name "Default Password Policy" \
  --set "last-login-time-attribute:ds-pwp-last-login-time" \
  --set "last-login-time-format:yyyyMMdd" \
  --set "last-login-time-zone:EST"\
  --set "previous-last-login-time-format:yyyyMMdd"
  --set "previous-last-login-time-attribute:ds-pwp-last-login-time" \
```

30.6.1.3 Configuring Password History Count and Duration

The `password-history-count` property specifies the number of past passwords that should be maintained in the history. A value of zero indicates that the server does not maintain a password history.

The `password-history-duration` property specifies the maximum length of time that a previously used password should remain in the user's password history. A value of `0 seconds` indicates that the server should not maintain a password history.

The following command configures password history count and duration for the default password policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop \
  --policy-name "Default Password Policy" --set "password-history-count:3" \
  --set "password-history-duration:5 seconds"
```

30.6.2 Creating a New Password Policy

You can configure and store multiple password policies with different configuration options. When you set up a directory server instance, the instance uses the default password policy

and applies it to all user entries, except root users (for example, the `cn=Directory Manager` account).

You can change the default password policy or you can create new password policies for specific groups in your directory. If a specific property is not present in a password policy, the server reads that property from the default password policy, in other words, all password policies inherit their default values from the default password policy.

The following command creates a new password policy and sets the `default-password-storage-scheme`, `lockout-duration`, `lockout-failure-count`, and `password-change-requires-current-password` properties. The remaining properties are inherited from the default Password Policy.

Use the `dsconfig` command to create a new password policy, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-password-policy \
  --policy-name "Temp Password Policy" --set password-attribute:userPassword \
  --set default-password-storage-scheme:"Salted SHA-1" \
  --set lockout-duration:300s --set lockout-failure-count:3 \
  --set password-change-requires-current-password:true
```

For more information about these properties, see Working with the Default Password Policy Properties.

30.6.3 Creating a First Login Password Policy

The First Login Password Policy is a specialized password policy that requires a user to change his password when first logging in to the system. Typically, an administrator sets up a new temporary password for newly created accounts, and the user is required to create his password after first logging in with the temporary password.

Use the `dsconfig` command to create a first login password policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
create-password-policy --policy-name "First Login Password Policy" \
--set password-attribute:userpassword \
--set default-password-storage-scheme:"Salted SHA-1" \
--set allow-user-password-changes:true \
--set force-change-on-add:true \
--set force-change-on-reset:true \
--set expire-passwords-without-warning:false \
--set password-expiration-warning-interval:"1 days" \
--set min-password-age:"0 seconds" \
--set max-password-age:"3 days" \
--set lockout-duration:"1 hours" \
--set lockout-failure-count:3 \
--set password-change-requires-current-password:true
```

For more information about these properties, see Working with the Default Password Policy Properties.

30.6.4 Assigning a Password Policy to an Individual Account

Assign a password policy to an individual by adding the `ds-pwp-password-policy-dn` attribute to the user's entry. The server then uses the configured password policy for that user.

1. Use `ldapmodify` to add the `ds-pwp-password-policy-dn` attribute.

```
$ ldapmodify --h localhost -p 1389 -D "cn=Directory Manager" \
-j pwd-file -X -n \
dn: uid=mgarcia,ou=Contractors,dc=example,dc=com
changetype: modify
add: ds-pwp-password-policy-dn
ds-pwp-password-policy-dn: cn=Temp Password Policy,cn=Password Policies,cn=config
```

2. Verify the entry by using `ldapsearch`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file -X -n \
  -b "dc=example,dc=com" -s sub "(uid=mgarcia)" ds-pwp-password-policy-dn
```

30.6.5 Preventing Password Policy Modifications

You must add an Access Control Instruction (ACI) to the root entry to prevent users from modifying their password policy.

Use the `ldapmodify` command with the specific ACI.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file -X -n \
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr != "ds-pwp-password-policy-dn")(version 3.0; acl "Allow self
modification except for ds-pwp-password-policy-dn";
allow (write) (userdn = "ldap:///self");)
```

30.6.6 Assigning a Password Policy to a Group of Users

You can assign a password policy to a group of users by adding a virtual attribute that automatically assigns the `ds-pwp-password-policy-dn` attribute to all the existing user entries that match the criteria associated with that virtual attribute. The criteria can be based entirely or in part on the group membership for a user.

Use `dsconfig` to create a virtual attribute that adds a password policy to a group of users.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-virtual-attribute \
  --name "Add PWPolicy to Admins" --type user-defined --set enabled:true \
  --set attribute-type:ds-pwp-password-policy-dn \
  --set group-dn:cn=Admins,ou=Groups,dc=example,dc=com \
  --set conflict-behavior:real-overrides-virtual \
  --set value:"cn=Admins PWPolicy,cn=Password Policies,cn=config"
```

30.6.7 Defining a Password Policy as an LDAP Subentry

LDAP subentries are special entries that hold operational data for the server. They are similar to operational attributes in that they are not returned to clients unless explicitly requested by including a Subentries Control request control.

You can define a password policy as an LDAP subentry, which means that the password policy is stored along with the user data, and can therefore be replicated.

Subentry password policies override the default password policy that is defined in the configuration. Settings that are not included in the subentry password policy are inherited from the default password policy.

When more than one password policy is defined under the same parent node with overlapping scope, the election of the password policy subentry that will apply to an entry within that scope

cannot be determined. You must therefore ensure that the password policies are defined in such a way that they do not conflict with each other.

Subentry password policies must rely on standard password policy properties only. A subentry password policy cannot contain password policy extension that are specific to Oracle Unified Directory.

To define a subentry password policy, create the password policy in an LDIF file, and add it to the data by using `ldapmodify`. You can specify the entries to which the password policy should be applied by including an LDAP filter in the subentry subtree specification.

The following example creates a password policy that applies only to a group of administrators. This password policy specifies the following:

- The user's account will be locked after a three successive failed password attempts.
- A failure interval of 300 seconds, after which a previously failed authentication attempt is no longer counted toward a lockout failure.
- A lockout duration of 300 seconds, after which it is automatically unlocked.
- Users to which this password policy applies can change their own passwords.
- Users with this password policy must change their password in a secure manner that does not expose the credentials.

1. Create an LDIF file (`admin-pwp.ldif`) that includes the entry specifying the password policy.

```
dn: cn=Admins Password Policy,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: pwdPolicy
cn: Admins Password Policy
pwdAttribute: userPassword
pwdLockout: TRUE
pwdMaxFailure: 3
pwdFailureCountInterval: 300
pwdLockoutDuration: 300
pwdAllowUserChange: TRUE
pwdSafeModify: TRUE
subtreeSpecification: {relativeBase "ou=people", specificationFilter
  "(isMemberOf=cn=Admins,ou=Groups,dc=example,dc=com)" }
```

2. Use the `ldapmodify` command to add the entry to the directory.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename admin-pwp.ldif
Processing ADD request for cn=Admins Password Policy,dc=example,dc=com
ADD operation successful for DN cn=Admins Password Policy,dc=example,dc=com
```

30.6.8 Deleting a Password Policy

You can delete any password policy, except the default password policy and the Default Root User Policy, from the directory when it is no longer needed.

In practice, first check the users who have the password policy you plan to delete, move them to a new password policy, and then remove the old password policy. If a password policy is deleted, any users who have a deleted password policy continue to have the `ds-pwd-password-policy-dn` pointing to the old password policy. The server returns an error when any requests to access the entry occur.

Use `dsconfig` to delete a password policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  delete-password-policy --policy-name "Temp Password Policy"
```

30.7 Managing Password Policies Using OUDSM

Use Oracle Unified Directory Services Manager (OUDSM) to manage the existing password policies and to modify the password policy properties.

The topics below provide step-by-step information to manage password policies using OUDSM:

- Listing the Configured Password Policy Subentries
- Creating a Password Policy Subentry
- Creating a Password Policy Subentry Based on an Existing Password Policy Subentry
- Deleting a Password Policy Subentry
- Displaying the Configured Password Policies
- Modifying a Password Policy
- Creating a Password Policy
- Creating a Password Policy Based on an Existing Password Policy
- Deleting a Password Policy
- Displaying the Supported Password Storage Schemes
- Enabling or Disabling a Password Storage Scheme

30.7.1 Listing the Configured Password Policy Subentries

Use Oracle Unified Directory Services Manager (OUDSM) to display all password policy subentries that are configured in the server.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy Subentry** element.

   The DNs of all password policy subentries are listed.
4. To display the details of a password policy subentry, select its DN.

   The password policy subentry properties are displayed in the right hand pane.
5. To modify any aspect of the password policy subentry, change the required value and click **Apply**.

For a description of all possible properties and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.

30.7.2 Creating a Password Policy Subentry

Use Oracle Unified Directory Services Manager (OUDSM) to create a new password policy subentry.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.

2. Select the **Security** tab.
3. Expand the **Password Policy Subentry** element.
4. Click the **Add** icon.

   The password policy subentry properties are displayed in the right hand pane.
5. On the **Create new password policy subentry** screen, complete the required fields.

   For a description of all possible properties, and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.
6. When you have completed configuring the password policy subentry, click **Create**.

30.7.3 Creating a Password Policy Subentry Based on an Existing Password Policy Subentry

Use Oracle Unified Directory Services Manager (OUDSM) to create a new password policy subentry that is based on an existing password policy subentry.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy Subentry** element.
4. Select the password policy subentry on which you want to base the new subentry.
5. Click the **Add like** icon.

   The properties of the original password policy subentry are displayed in the right hand pane.
6. Modify the required values.

   For a description of all possible properties, and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.
7. When you have completed configuring the new password policy subentry, click **Create**.

30.7.4 Deleting a Password Policy Subentry

Use Oracle Unified Directory Services Manager (OUDSM) to delete a password policy subentry.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy Subentry** element.
4. Select the password policy subentry that you want to deleted.
5. Click the **Delete** icon.

   You are prompted to confirm the deletion. Click **OK**.

30.7.5 Displaying the Configured Password Policies

Use Oracle Unified Directory Services Manager (OUDSM) to display the list of password policies.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.

   The list of configured password policies is displayed.
4. Select a password policy to display its properties in the right hand pane.

For a description of all possible properties and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.

30.7.6 Modifying a Password Policy

Use Oracle Unified Directory Services Manager (OUDSM) to modify a configured password policy.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.

   The list of configured password policies is displayed.
4. Select the password policy whose properties you want to modify.

> **Note:**
>
> - You can also use OUDSM to modify the Default Password Policy. See Modifying the Default Password Policy for more information.
> - For a description of all possible password policy properties, and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.

30.7.7 Creating a Password Policy

Use Oracle Unified Directory Services Manager (OUDSM) to create a new password policy.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.
4. Click the **Add** icon.
5. On the **Create New Password Policy** screen, configure the required properties.

   For a description of all possible properties, and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.
6. When you have configured the new password policy, click **Create**.

30.7.8 Creating a Password Policy Based on an Existing Password Policy

Use Oracle Unified Directory Services Manager (OUDSM) to create a new password policy that is based on an existing password policy.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.
4. Select the password policy on which you want to base the new policy.
5. Click the **Add like** icon.
6. On the **Create New Password Policy** screen, modify the properties to create the new policy.

   For a description of all possible properties, and their values, see "Password Policy" in the *Configuration Reference for Oracle Unified Directory*.
7. When you have configured the new password policy, click **Create**.

30.7.9 Deleting a Password Policy

Use Oracle Unified Directory Services Manager (OUDSM) to delete a password policy.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.
4. Select the password policy that you want to delete.
5. Click the **Delete** icon.
6. Click **OK** to confirm the deletion.

30.7.10 Displaying the Supported Password Storage Schemes

A password storage scheme provides a mechanism for encoding user passwords for storage in the server. In most cases, the password is encoded in a manner that prevents users from determining what the clear-text password is, while still allowing the server to determine whether the user-supplied password is correct.

Oracle Unified Directory supports several password storage schemes. See password storage scheme.

To display the list of password storage schemes using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Storage** element.
4. The list of password storage schemes is displayed.

30.7.11 Enabling or Disabling a Password Storage Scheme

You can use Oracle Unified Directory Services Manager to enable or disable a password storage scheme.

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Storage** element.
4. Select the password storage scheme that you want to enable or disable.
5. In the right hand pane, check or uncheck the **Enabled** box, as required.
6. Click **Apply** to save your changes.

30.8 Managing Password Validators

Password validators provide a mechanism to determine whether a provided plain text password is acceptable for use. Validation prevents users from choosing trivial passwords that are weak and might be easily guessed.

Types of validation that might be performed include:

- Ensuring that a password has at least a specified minimum number of characters.
- Ensuring that a password has no more than a specified maximum number of characters.
- Ensuring that a password contains at least a specified number of characters from different character sets (for example, lowercase letters, uppercase letters, numeric digits, and symbols).
- Ensuring that a user is not allowed to reuse a password that has been previously used by that user (that is, that the password is not contained in a history of previous passwords).
- Ensuring that a user is not allowed to choose a password that matches the value of another attribute in the user's entry.
- Ensuring a password is not contained in a specified dictionary.

The password policy for a user specifies the set of password validators that should be used whenever that user provides a new password. To activate a password validator, you must enable the corresponding configuration entry, and include the DN of that entry in the `password-validator` attribute of the password policy in which you want that validator active.

The following password validators are available in the server by default:

- **Attribute Value Password Validator**

  This validator attempts to determine whether a proposed password is acceptable for use by determining whether that password is contained in any attribute within the user's entry.You can configure the validator to look in all attributes or in a specified subset of attributes.

- **Character Set Password Validator**

  This validator determines whether a proposed password is acceptable by checking whether it contains enough characters from one or more user-defined character sets.For example, the validator can ensure that passwords must have at least one lowercase letter, one uppercase letter, one digit, and one symbol.

This validator also ensures that a proposed password contains characters from a minimum number of character sets (with `use-any-of` property) rather than characters from all configured character sets. For example, if four character sets are configured and the `use-any-of` property is set to 3, proposed passwords must contain characters from at least three of the four character sets. If users prefer, passwords can also contain characters from all four of the configured character sets.

See the example in Configuring the Values of a Password Validator.

- **Dictionary Password Validator**

  This validator determines whether a proposed password is acceptable based on whether the password value appears in a provided dictionary file.A large dictionary file is provided with the server, but you can supply an alternate dictionary. In this case, the dictionary must be a plain-text file with one word per line.

- **Length Based Password Validator**

  This validator determines whether a proposed password is acceptable based on whether the number of characters it contains falls within an acceptable range of values.Both upper and lower bounds can be defined.

- **Repeated Characters Password Validator**

  This validator determines whether a proposed password is acceptable based on the number of times any character appears consecutively in a password value.It ensures that user passwords do not contain strings of the same character repeated several times, like "aaaaaa" or "aaabbb".

- **Similarity Based Password Validator**

  This validator determines whether a proposed password is acceptable by measuring how similar it is to the user's current password.In particular, it uses the Levenshtein Distance algorithm to determine the minimum number of changes (where a change may be inserting, deleting, or replacing a character) to transform one string into the other. It can be used to prevent users from making only minor changes to their current password when setting a new password.

  > **Note:**
  >
  > For this password validator to be effective, it must have access to the user's current password. Therefore, to enable this password validator, the `password-change-requires-current-password` property in the password policy configuration must also be set to `true`.

- **Unique Characters Password Validator**

  This validator determines whether a proposed password is acceptable based on the number of unique characters that it contains.It can be used to prevent simple passwords that contain only a few characters like "aabbcc" or "abcabc".

30.8.1 Managing Password Validators by Using the Command Line

Use the `dsconfig` command to manage password validators and their properties.

The following topics provide a step-by-step information to manage password validators by using the `dsconfig` command:

- Displaying the Available Password Validators

- Displaying the Properties of a Password Validator
- Enabling or Disabling a Password Validator
- Configuring the Values of a Password Validator
- Associating a Password Validator With a Password Policy
- Defining a Password Validator as an LDAP Subentry

30.8.1.1 Displaying the Available Password Validators

To view a list of available password validators:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-password-validators
Password Validator                  : Type                : enabled
------------------------------------:---------------------:--------
Attribute Value                     : attribute-value     : true
Character Set                       : character-set       : true
Dictionary                          : dictionary          : false
Length-Based Password Validator     : length-based        : true
Repeated Characters                 : repeated-characters : true
Similarity-Based Password Validator : similarity-based    : true
Unique Characters                   : unique-characters   : true
```

30.8.1.2 Displaying the Properties of a Password Validator

To view the properties of a password validator:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-password-validator-prop --validator-name "Length-Based Password Validator"
Property            : Value(s)
--------------------:---------
enabled             : true
max-password-length : 0
min-password-length : 8
```

30.8.1.3 Enabling or Disabling a Password Validator

All of the password validators, except the Dictionary validator, are enabled by default. You must enable a validator before it can be associated with a specific password policy.

Use the `dsconfig` command to set the `enabled` property to `true` or `false`. For example, to disable the Length-Based password validator, set the `enabled` property as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-validator-prop --validator-name "Length-Based Password Validator" \
  --set enabled:false
```

30.8.1.4 Configuring the Values of a Password Validator

Use the `dsconfig` command to configure properties of a password validator. For example, to specify that passwords must be at least eight characters long, set the `min-password-length` property as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-validator-prop --validator-name "Length-Based Password Validator" \
  --set min-password-length:8
```

To specify that passwords must contain characters from at least three of four configured character sets, use `dsconfig`, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-validator-prop --validator-name "Character Set" \
  --set enabled:true
  --set allow-unclassified-characters:false
  --set character-set:3:ABCDEFGHIJKLMNOPQRSTUVWXYZ
  --set character-set:3:abcdefghijklmnopqrstuvwxyz
  --set character-set:2:0123456789
  --set character-set:2:~!@#$%^&*()-_=+[]{}|;:,.<>/?
  --set use-any-of:3
```

In this example, passwords can also contain characters from all four of the configured character sets, if users prefer.

30.8.1.5 Associating a Password Validator With a Password Policy

A password validator is only taken into account when it is associated with a specific password policy.

To associate a password validator with a password policy, set the `password-validator` property of the password policy.

For example, to specify that the default password policy should check whether passwords conform to a specific number of characters, set the `password-validator` property of the default password policy as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop --policy-name "Default Password Policy" \
  --set password-validator:"Length-Based Password Validator"
```

30.8.1.6 Defining a Password Validator as an LDAP Subentry

LDAP subentries are special entries that hold operational data for the server. They are similar to operational attributes in that they are not returned to clients unless explicitly requested by including a Subentries Control request control.

You can define a password validator as an LDAP subentry, which means that the password validator is stored along with the user data, and can therefore be replicated. Subentry password validators can be attached only to Subentry Password Policies.

We can have any number of Subentry Password Validators under the same parent, We need to specify the exact DN while mapping it to a subentry password policy. If no password validator is attached to a subentry password policy it will inherit the validators configured to the Default Password Policy.

To define a subentry password validator, create the password validator in an LDIF file (length-based.ldif), and add it to the data by using ldapmodify.

The following example creates a Length-Based password validator with the following properties. The maximum password length allowed is 25 characters. The minimum password length allowed is 10 characters.

1. Run the following command:

```
dn: cn=LengthBasedSubentryPV,ou=people,dc=example,dc=com
changeType: add
objectClass: top
objectClass: ds-cfg-password-validator
```

```
objectClass: ds-cfg-length-based-password-validator
objectClass: subentry
ds-cfg-enabled: true
ds-cfg-max-password-length: 25
cn: Length-Based Subentry PV
ds-cfg-java-class:
org.opends.server.extensions.LengthBasedPasswordValidator
ds-cfg-min-password-length: 10
subtreeSpecification: {}
```

> **Note:**
>
> Leave the subtreeSpecification empty, this attribute value will not be taken into account for Password Validators.

2. Use the `ldapmodify` command to add the entry to the directory.

```
ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
 --defaultAdd --filename length-based.ldif
Processing ADD request for
cn=LengthBasedSubentryPV,ou=people,dc=example,dc=com
ADD operation successful for DN
cn=LengthBasedSubentryPV,ou=people,dc=example,dc=com
```

3. Map the above created password validator to a subentry password policy by creating the following LDIF file `map-pwp-validator.ldif`.

```
dn: cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
changeType: modify
add: objectClass
objectClass: oudPwdPolicyAdvanced
-
add: ds-cfg-password-validator
ds-cfg-password-validator:
cn=LengthBasedSubentryPV,ou=people,dc=example,dc=com
```

4. Use the `ldapmodify` command to add the entry to the directory.

```
ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
 -a -f map-pwp-validator.ldif
Processing MODIFY request for
cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
MODIFY operation successful for DN
cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
```

Similarly, you can perform modify operations to the subentry password validators using `ldapmodify`. OUD will perform the referential Integrity checks for the delete operations of subentry password validators. OUD will throw an error if the password validator have been referenced by any of the Subentry Password Policy.

30.8.2 Managing Password Validators Using OUDSM

Use Oracle Unified Directory Services Manager (OUDSM) to manage password validators and their properties.

The following topics provide a step-by-step information to manage password validators by using the OUDSM interface:

- Displaying the Available Password Validators
- Displaying the Properties of a Password Validator
- Enabling or Disabling a Password Validator
- Configuring the Properties of a Password Validator
- Associating a Password Validator With a Password Policy

30.8.2.1 Displaying the Available Password Validators

To view a list of available password validators:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Validator** element.

   The available password validators are displayed.

30.8.2.2 Displaying the Properties of a Password Validator

To display the properties of a password validator:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Validator** element.

   The available password validators are displayed.
4. Click a password validator to display its properties in the right hand pane.

30.8.2.3 Enabling or Disabling a Password Validator

All of the password validators, except the Dictionary validator, are enabled by default. You must enable a validator before it can be associated with a specific password policy.

To enable or disable a password validator:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Validator** element.

   The available password validators are displayed.
4. Click a password validator to display its properties in the right hand pane.

5. Select the **Enabled** check box to enable the validator, or deselect this check box to disable the validator.
6. Click **Apply** to save the configuration changes.

30.8.2.4 Configuring the Properties of a Password Validator

To configure the properties of a password validator by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Validator** element.

   The available password validators are displayed.
4. Click a password validator to display its properties in the right hand pane.
5. Configure any required properties and click **Apply** to save the configuration change.

30.8.2.5 Associating a Password Validator With a Password Policy

A password validator is only taken into account when it is associated with a specific password policy.

To associate a password validator with a password policy:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Security** tab.
3. Expand the **Password Policy** element.

   The available password policies are displayed.
4. Click a password policy to display its properties in the right hand pane.
5. Expand the Syntax element in the right hand pane.
6. From the **Password Validator** list, select the password validators that you want to associate with this password policy.
7. Click **Apply** to save the configuration changes.

30.9 Managing Password Generators

Password generators are used to generate passwords for user accounts. A password generator is used with the password modify extended operation to provide a new password for cases in which the client did not include a password in its request.

If no password generator is associated with the password policy that is in force, the password modify extended operation does not automatically generate passwords for users.

The passwords that are created by a password generator are not subject to validation. You should configure password generators so that the passwords they create are in-line with the requirements of the associated password validators.

By default one password generator is configured on a directory server instance - the random password generator. The following sections describe how to manage password generators by using `dsconfig`:

30.9.1 Displaying the Configured Password Generators

Use the `dsconfig` command to list the configured password generators.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-password-generators
Password Generator        : Type   : enabled
--------------------------:--------:--------
Random Password Generator : random : true
```

30.9.2 Displaying the Properties of a Password Generator

Use the `dsconfig` command to display the properties of a password generator.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-password-generator-prop --generator-name "Random Password Generator"
Property               : Value(s)
-----------------------:---------------------------------------------------
enabled                : true
password-character-set : alpha:abcdefghijklmnopqrstuvwxyz, numeric:0123456789
password-format        : "alpha:3,numeric:2,alpha:3"
```

The password character set is a multi-valued property, with each value defining a different character set. The format of the character set is the name of the set followed by a colon and the characters that are in that set. For example, the value "alpha:abcdefghijklmnopqrstuvwxyz" defines a character set named "alpha" containing all of the lower-case ASCII alphabetic characters.

The password format is a comma-delimited list of elements in which each of those elements consists of the name of a character set defined in the `password-character-set` property, a colon, and the number of characters to include from that set. For example, the default value of "`alpha:3,numeric:2,alpha:3`" generates an 8-character password in which the first three characters are from the "alpha" set, the next two are from the "numeric" set, and the final three are from the "alpha" set.

30.9.3 Enabling or Disabling a Password Generator

The random password generator is enabled by default. A validator must be enabled before it can be associated with a specific password policy. Use the `dsconfig` command to set the `enabled` property to `true` or `false`.

For example, to disable the random password generator, set the `enabled` property as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-generator-prop --generator-name "Random Password Generator" \
  --set enabled:false
```

30.9.4 Configuring the Properties of a Password Generator

Use the `dsconfig` command to configure properties of a password generator.

For example, to specify that passwords generated by the random password generator must be of the form, three letters, three numbers, and two defined special characters, set the corresponding properties as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-generator-prop --generator-name "Random Password Generator" \
  --add password-character-set:special:\!@#\$%^&*\(\)
  --set password-format:alpha:3,numeric:3,special:2
```

30.9.5 Associating a Password Generator With a Password Policy

A password generator is only taken into account when it is associated with a specific password policy. Set the `password-generator` property of the password policy to associate a password generator with a password policy by using `dsconfig`.

For example, to specify that the default password policy should use a new password generator, named `Special Generator`, set the `password-generator` property of the default password policy as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-password-policy-prop --policy-name "Default Password Policy" \
  --set password-generator:"Special Generator"
```

30.9.6 Defining a Password Generator as an LDAP Subentry

LDAP subentries are special entries that hold operational data for the server. They are similar to operational attributes in that they are not returned to clients unless explicitly requested by including a Subentries Control request control.

You can define a password generator as an LDAP subentry, which means that the password generator is stored along with the user data, and can therefore be replicated. Subentry password generator can be attached only to Subentry Password Policies.

You can have any number of Subentry Password generator under the same parent. You need to specify the exact DN while mapping it to a subentry password policy. If no password generator is attached to a subentry password policy it will inherit the generator configured to the Default Password Policy.

To define a subentry password generator, create the password generator in an LDIF file (length-based.ldif), and add it to the data by using ldapmodify.

1. The following example creates a random password generator:

```
dn: cn=RandomPassGenerator,ou=people,dc=example,dc=com
changetype: add
objectClass: ds-cfg-random-password-generator
objectClass: top
objectClass: ds-cfg-password-generator
objectClass: subentry
ds-cfg-enabled: true
ds-cfg-password-format: alpha:3,numeric:2,alpha:3
cn: RandomPassGenerator
ds-cfg-java-class: org.opends.server.extensions.RandomPasswordGenerator
```

```
ds-cfg-password-character-set: alpha:abcdefghijklmnopqrstuvwxyz
ds-cfg-password-character-set: numeric:0123456789
subtreeSpecification: {}
```

> **Note:**
>
> Leave the subtreeSpecification empty, this attribute value will not be taken into account for Password Validators.

2. Use the `ldapmodify` command to add the entry to the directory.

```
ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
 --defaultAdd --filename random-generator.ldif
Processing ADD request for
cn=RandomPassGenerator,ou=people,dc=example,dc=com
ADD operation successful for DN
cn=RandomPassGenerator,ou=people,dc=example,dc=com
```

3. Map the above created password generator a subentry password policy by creating the following LDIF file `map-pwp-generator.ldif`.

```
dn: cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
changeType: modify
add: objectClass
objectClass: oudPwdPolicyAdvanced
-
add: ds-cfg-password-generator
ds-cfg-password-generator:
cn=RandomPassGenerator,ou=people,dc=example,dc=com
```

4. Use the `ldapmodify` command to add the entry to the directory.

```
ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
 -a -f map-pwp-generator.ldif
Processing MODIFY request for
cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
MODIFY operation successful for DN
cn=subEntryPasswordPolicy,ou=people,dc=example,dc=com
```

Similarly, you can perform modify operations to the subentry password generator using ldapmodify. OUD will perform the referential Integrity checks for the delete operations of subentry password generator. OUD will throw an error if the password generator have been referenced by any of the Subentry Password Policy.

> **Note:**
>
> The OUDSM support for Subentry Password Validator and Subentry Password Generator is not available.

31

Integrating Oracle Unified Directory with Oracle Enterprise User Security

Oracle Enterprise User Security (EUS) enables Oracle Database users to authenticate against identities stored in an LDAP-compliant directory service.
Topics:

31.1 Understanding How Oracle Enterprise User Security Works with Oracle Unified Directory

Oracle Enterprise User Security enables you to centrally manage database users across the enterprise. You can create enterprise users in an LDAP-compliant directory service, and then assign roles and privileges across various enterprise databases registered with the directory.

Users connect to Oracle Database by providing credentials stored in Oracle Unified Directory or other external LDAP-compliant directory front-ended by Oracle Unified Directory proxy server. The database executes LDAP search operations to query user specific authentication and authorization information. For more information, see Configuration 6: Enterprise User Security.

Integrating Oracle Unified Directory and Enterprise User Security enhances and simplifies your authentication and authorization capabilities by allowing you to leverage user identities stored in LDAP-compliant directory service without any additional synchronization.

For more information about Oracle Enterprise User Security, see the *Oracle Database Enterprise User Security Administrator's Guide*.

31.2 Understanding the Options Before Integrating Oracle Unified Directory with Oracle Enterprise User Security

Before you integrate Oracle Unified Directory with Oracle Enterprise User Security, you should consider what role Oracle Unified Directory will play in your topology. Also consider other business requirements for your enterprise.

Before you begin integration, review all tasks and steps required for the various integration options.

- Is OUD used as a directory server or as a directory proxy in the topology?

  When you use OUD as a directory server, installation is straightforward, and configuration is contained in OUD. For more information, see Configuring Oracle Directory Server as a Directory for Enterprise User Security.

  When you use OUD as a directory proxy, you must take additional steps to configure the external LDAP-compliant directory that stores user entries. For more information, see Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security.

- Are you configuring an existing directory or proxy instance, or installing a new instance?

  If you are configuring an existing directory or proxy instance to work with Enterprise User Security, you will need to complete some configuration steps manually. See the following for more information:

  – Configuring Oracle Unified Directory to Work with Enterprise User Security
  – Configuring User Identities in the External LDAP Directory

  If you are installing a new directory or proxy instance, you can choose the Enterprise User Security option during setup. The new instance is automatically configured to EUS integration. See the following for more information:

  – Installing and Configuring a New Oracle Unified Directory Instance to Work with Enterprise User Security
  – Configuring Oracle Unified Directory Proxy to Work with Enterprise User Security

- Additional business requirements for you to consider.

  See the following for more information:

  – Configuring OUD to Support Multiple Enterprise User Security Domains
  – Using Oracle Unified Directory and Enterprise User Security in High Availability Topologies

31.3 About the Prerequisites Before Integrating Oracle Unified Directory with Oracle Enterprise User Security

Make sure you review the prerequisites before integrating Oracle Unified Directory with multiple Oracle products, as well as any external LDAP-compliant directory you may have in your topology.

Before you begin, ensure that you can access the following components as well as the current documentation that goes with them:

- Oracle Unified Directory, OUDSM, `oud-setup` and `oud-proxy-setup` commands
- Oracle Enterprise User Security Net Configuration Assistant
- Database Configuration Assistant for Oracle Database
- Enterprise Manager for Oracle Database
- Supported LDAP directories (Microsoft Active Directory, Novell eDirectory, Oracle Unified Directory, or Oracle Directory Server Enterprise Edition) you have in your topology

31.4 Enabling Oracle Unified Directory and Oracle Enterprise User Security to Work Together

Follow these step-by-step instructions for integrating Oracle Unified Directory with Oracle Enterprise User Security.

- Configuring Oracle Directory Server as a Directory for Enterprise User Security
- Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security
- Configuring Password Policy for Oracle Unified Directory Administrator

31.4.1 Configuring Oracle Directory Server as a Directory for Enterprise User Security

Follow these tasks to configure Oracle Unified Directory Server as a directory for Enterprise User Security.

To configure Oracle Directory Server as a directory for Enterprise User Security, complete the tasks described in the following table:

Table 31-1 List of Tasks to configure Oracle Directory Server as a directory for Enterprise User Security

Task #	Link
Task 1	Configuring Oracle Unified Directory to Work with Enterprise User Security
Task 2	Configuring the User and Groups Location
Task 3	Selecting the Oracle Context to be Used by Enterprise User Security
Task 4	Registering the Database in the LDAP Server
Task 5	Configuring Roles and Permissions
Task 6	Testing the Database Configurations

31.4.1.1 Configuring Oracle Unified Directory to Work with Enterprise User Security

- If you already have an existing Oracle Unified Directory instance installed and provisioned, then complete the steps in one of these sections:
  - Configuring an Existing Oracle Unified Directory Server to Work with Enterprise User Security Using the Command Line
  - Configuring an Existing Oracle Unified Directory Server to Work with Enterpriser User Security Using OUDSM
- If you do not already have an Oracle Unified Directory installed and provisioned, then complete the steps in the following section Installing and Configuring a New Oracle Unified Directory Instance to Work with Enterprise User Security

31.4.1.1.1 Installing and Configuring a New Oracle Unified Directory Instance to Work with Enterprise User Security

You can run the `oud-setup` program using either the command line or the graphical user interface.

- To run `oud-setup` with following `--cli` option. For example:

```
$ oud-setup --cli --integration eus --no-prompt --ldapPort 1389\
 --adminConnectorPort 4444 -D "cn=directory manager"\
 --rootUserPasswordFile pwd.txt --ldapsPort 1636\
 --generateSelfSignedCertificate --baseDN "dc=example,dc=com"
```

  For detailed information about using `oud-setup` and all its options, see "Setting Up the Directory Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Unified Directory*

  During setup, the `baseDN` specified in the `--baseDN` option is prepared for EUS. If you specify multiple base DNs, they will all be prepared for EUS.

  Using the above command, you can configure OUD instance to use Salted SHA-1 password storage scheme. However, you can configure OUD to use the more secure EUS PBKDF2 SHA512 password storage scheme, which encodes password using SHA-512 based algorithm.

  In order to do so, run the `oud-setup` command using `eusPasswordScheme` argument with value "`sha2`". For example:

```
oud-setup --cli --integration eus --no-prompt --ldapPort 1389\
 --adminConnectorPort 4444 -D "cn=directory manager"\
 --rootUserPasswordFile pwd.txt --ldapsPort 1636\
 --generateSelfSignedCertificate --baseDN "dc=example,dc=com" --
eusPasswordScheme sha2
```

  > **Note:**
  >
  > – You can configure Oracle Unified Directory to use `EUS PBKDF2 SHA512` password storage scheme only if your Oracle RDBMS version supports it. Oracle recommends that you contact your Database Administrator to validate if the RDBMS supports `Multi-Round SHA-512` based password verifier.
  >
  > – You can configure OUD to use `EUS PBKDF2 SHA512` password storage scheme only using CLI option. The same is not supported in GUI mode.

- To use the graphical user interface:
  1. Run the `oud-setup` command
  2. In the Welcome page, click **Next**.
  3. In the Server Settings page, provide the following information:
     a. **Host Name**

This is the server that hosts the Oracle Unified Directory instance that stores users and groups.

b. **Administration Connector Port**

This is the administration port used by OUD tools such as dsconfig.

c. **LDAP Listener Port**

Specify the port used by OUD.

d. **LDAP Secure Access**

Click Configure to enable secure access.

In the Configure Secure Access window, click to mark the Enable SSL on Port check box. Then enter a port number for LDAPS, and click **OK** to continue.

e. **Root User DN**

This is the identity of the server administrator

f. **Password**

Enter a password to be used by the server administrator.

g. **Password (confirm)**

Enter the password a second time to confirm.

Click Next to continue.

4. In the Topology Options page, be sure the option "This will be a stand alone server" is selected, and click **Next**.
5. In the Directory Data page, provide the following information:

a. Directory Base DN

Enter the base DN where you will store user entries.

b. Directory Data

Do not choose the option "Leave Database Empty." Choose one of the following options:

- "Only Create Base Entry" creates an entry with the base DN specified previously.
- "Import Data from LDIF File" imports LDIF data from the file specified in the Path field.
- "Import Automatically-Generated Sample Data" generates the number of sample entries specified in the Number of User Entries field.

Click **Next**.

6. In the Oracle Components Integration page, choose the option "Enable for EUS (Enterprise User Security), EBS, Database Net Services and DIP." This option also enables the server for Database Net Services.

Click **Next** to continue.

7. In the Server Tuning page, you can configure your tunings or click **Next**.

See the Installation Guide for information about tuning configurations.

8. In the Review page, review your settings, and click **Finish**.

A new instance of Oracle Unified Directory is installed, configured, and then started.

31.4.1.1.2 Configuring an Existing Oracle Unified Directory Server to Work with Enterprise User Security Using the Command Line

You can configure an existing naming context for EUS, or you can create and configure a new naming context for EUS.

- To use an existing naming context for EUS, run the `manage-suffix update` command. For example:

  ```
  $ manage-suffix update -h host -p adminPort -D "cn=directory manager" -j
  pwd.txt -X -n -b baseDN --integration eus
  ```

  This command-line will configure the naming context specified as `baseDN` for EUS.

  Using the above command, you can configure OUD to use Salted SHA-1 password storage scheme. However, you can configure OUD to use the more secure EUS PBKDF2 SHA512 password storage scheme, which encodes password using SHA-512 based algorithm. In order to do so, run the `manage-suffix update` command using eusPasswordScheme argument with value "sha2". For example:

  ```
  $ manage-suffix update -h host -p adminPort -D "cn=directory manager" -j
  pwd.txt -X -n -b baseDN --integration eus
  --eusPasswordScheme sha2
  ```

- To create a new naming context for EUS, run the `manage-suffix create` command. For example:

  ```
  $ manage-suffix create -h host -p adminPort -D "cn=directory manager" -j
  pwd.txt -X -n -b baseDN --integration eus
  ```

  Using the above command, you can configure OUD to use Salted SHA-1 password storage scheme. However, you can configure OUD to use the more secure EUS PBKDF2 SHA512 password storage scheme, which encodes password using SHA-512 based algorithm. In order to do so, run the `manage-suffix create` command using eusPasswordScheme argument with value "sha2". For example:

  ```
  $ manage-suffix create -h host -p adminPort -D "cn=directory manager" -j
  pwd.txt -X -n -b baseDN --integration eus
  --eusPasswordScheme sha2
  ```

> **Note:**
>
> You can configure Oracle Unified Directory to use `EUS PBKDF2 SHA512` password storage scheme only if your Oracle RDBMS version supports it. Oracle recommends that you contact your Database Administrator to validate if the RDBMS supports `Multi-Round SHA-512` based password verifier.

For more information about the `manage-suffix` command, see Managing Suffixes Using `manage-suffix`.

31.4.1.1.3 Configuring an Existing Oracle Unified Directory Server to Work with Enterpriser User Security Using OUDSM

Before you begin, ensure that the server instance has an LDAP connection handler that is enabled for SSL. If SSL is not enabled, add an LDAPS connection handler. For information about adding an LDAPS connection handler, see Managing the Server Configuration Using `dsconfig`, and Displaying the Properties of LDAP Connection Handler.

You can configure an existing naming context for EUS, or you can create and configure a new naming context for EUS.

- To configure an existing naming context for EUS using OUDSM:
  1. Connect to the directory server from OUDSM.
  2. Click the **Configuration** tab.
  3. In the navigation pane on the left, below Naming Contexts, choose the naming context you want to use.
  4. In the right pane, in the Oracle Components Integration section, choose **Enable for Enterprise User Security (EUS)** and click **Apply**.
- To create and configure a new naming context for EUS using OUDSM:
  1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
  2. Click the **Home** tab.
  3. Under the Configuration menu, choose **Create Local Naming Context**.
  4. In the New Local Naming Context window, provide the following information:
     a. **Base DN**

        Type a name for the suffix that you want to create. You cannot enable EUS on an existing suffix that has already been populated with user data.

     b. **Directory Data Options**

        Choose one of the following:

        Only Create Base Entry creates the database along with the base entry of the suffix. Any additional entries must be added after suffix creation.

        Leave Database Empty creates an empty database. Do not select this option.

        When you use this option, the base entry and any additional entries must be added after suffix creation. But for this configuration, the suffix must contain at least one entry.

        Import Generated Sample Data populates the suffix with sample entries.

        Specify the number of entries that should be generated in the **Number of User Entries** field. You can import a maximum of 30,000 sample entries through OUDSM. If you want to add more than 30,000 entries, you must use the `import-ldif` command.

     c. **Oracle Components Integration**

        To enable the new suffix, for Enterprise User Security (EUS), select **Enable**.

     d. **Network Group**

        Attach the suffix to at least one network group:

To attach the suffix to an existing network group: Choose **Use Existing**, and then choose the required network group from the list.

To attach the suffix to a new network group: Select **Create New**, and then in the Name field, type a name for the network group you want to create.

You can attach the same suffix to several network groups.

e. **Workflow Element**

Attach the suffix to the workflow element.

To attach the suffix to an existing workflow element: Choose **Use Existing**, and then choose the required workflow element from the list.

The suffix is stored inside the same database Local Backend workflow element, and will have the same properties such as an instance path to Berkeley DB files.

To attach the suffix to a new workflow element: Choose **Create New**, and then in the Name field, type a name for the workflow element you want to create.

You can configure this new workflow element with additional other values such as Berkeley DB files, database cache size, and so on.

5. Click **Create**.

The following confirmation message is displayed:

```
Naming Context created successfully.
```

> **Note:**
>
> After creating and configuring a naming context for EUS, the Oracle Unified Directory configuration can be updated to enable the TNS Aliasing capability.
>
> You must manually run the `dsconfig` command to set up the TNS Aliasing feature by adding the `eus-alias-resolution` workflow element into the global `cn=OracleContext` and also the `cn=OracleContext,<EUS Realm>` workflow chains. See Enabling TNS Alias Support for EUS-enabled Configurations in *Installing Oracle Unified Directory*.

31.4.1.2 Configuring the User and Groups Location

After Oracle Unified Directory has been configured for EUS or Oracle E-Business Suite, you must configure the naming context used to store the users and the groups by performing the following steps:

1. Locate the LDIF template file at *`install_directory`*`/config/EUS/modifyRealm.ldif`.
2. Edit the `modifyRealm.ldif` file as follows:
   - Replace `dc=example,dc=com` with the correct naming context for your server instance.
   - Replace `ou=people` and `ou=groups` with the correct location of the user and group entries in your DIT.
3. Use the `ldapmodify` command to update the configuration with the edited LDIF template file, for example:

```
$ ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file -f
modifyRealm.ldif
```

> **Note:**
>
> Ensure that you specify the port number on which the LDAP Connection Handler will listen for connections from clients (For example, `1389`) and not the administration port number which is `4444`.

31.4.1.3 Selecting the Oracle Context to be Used by Enterprise User Security

Enterprise User Security stores its configuration, also called EUS metadata, in an Oracle Context which corresponds to a part of the Directory Information Tree. If your user entries are stored below `dc=example,dc=com`, then EUS is usually configured to use `cn=OracleContext,dc=example,dc=com` as Oracle Context.

Use Oracle Net Configuration Assistant to indicate where EUS should read its configuration.

1. To start the Oracle Net Configuration Assistant, run the `netca` command on the host where the database is installed.
2. On the Welcome page, select "Directory Usage Configuration," and click Next.

   On the subsequent pages, provide the following information:

   - **Directory Type**

     Select "Oracle Internet Directory" even if the LDAP server is an Oracle Virtual Directory or an Oracle Unified Directory.

     Click Next.

   - **Hostname**

     Enter the hostname or IP address of the server hosting your LDAP server.

   - **Port**

     Enter the LDAP port number.

   - **SSL Port**

     Enter the LDAPS port number.

   - **Oracle Context**

     Do not select `cn=OracleContext`. Instead, click the arrow to display and choose the location of your OracleContext.

     Click Next.
3. When the following message is displayed, click Next: "Directory usage configuration complete!"
4. When the Welcome page is displayed, click Finish.
5. To verify that the Net Configuration Assistant has successfully created the configuration file containing the LDAP server information, run the following command:

```
# cat $ORACLE_HOME/network/admin/ldap.ora
# ldap.ora Network Configuration File: /app/oracle/product/db/product/11.2.0/
dbhome_1/network/admin/ldap.ora
# Generated by Oracle configuration tools.
DIRECTORY_SERVERS= (oudhost:1389:1636)
DEFAULT_ADMIN_CONTEXT = "dc=example,dc=com"
DIRECTORY_SERVER_TYPE = OID
```

The configuration file used by the database contains the hostname and port of the LDAP server. In this example, the information is represented as: `(oudhost:1389:1636)`. You can specify multiple servers, separated by commas, for high availability deployments. See Using Oracle Unified Directory and Enterprise User Security in High Availability Topologies.

In this example, `dc=example,dc=com` represents the Oracle Context used to store the EUS configuration, also known as the EUS metadata.

31.4.1.4 Registering the Database in the LDAP Server

Use the Database Configuration Assistant for Oracle Database to complete this task.

1. Run the `dbca` command on the host where the database is installed.

   The Database Configuration Assistant for Oracle Database is displayed. Click Next, then provide the following information in the subsequent pages:

   - Select the operation you want to perform

     Choose "Configure Database Option," then click Next.

   - Database

     In the list box, select the database you want to register. Then click Next.

     Database Configuration Assistant determines if the database is already registered in the LDAP server.

   - Would you like to register this database with the directory service?

     Choose "Yes, register the database." Database Configuration Assistant will create an entry for the database in the Oracle Context.

   - User DN

     The user DN will be used to authenticate to the LDAP server. The user DN is also used in the add operation, which creates the database entry in the Oracle Context. The user must have write access to the LDAP server.

   - Password

     Database Configuration Assistant creates a wallet for the database. The database entry DN and password will be stored in the wallet. When the database connects to the LDAP server, it will authenticated using credentials stored in this wallet.

   - Database Components

     Make no changes to this page, and click Next.

   - Connection Mode

     Choose "Dedicated Server Mode," then click Finish.

   - Confirmation

     Click OK to register the database.

   - Do you want to perform another operation?

     Click No to exit the Database Configuration Assistant application.

2. To verify that Database Configuration Assistant successfully created a new entry for the database, run the following command, where **cn=orcl11g** is the name of the database specified in the previous step:

```
$ ldapsearch -h oudhost -p 1389 -D "cn=directory manager" -j pwd.txt -b
cn=oraclecontext,dc=example,dc=com "(cn=orcl11g)"
dn: cn=orcl11g,cn=OracleContext,dc=example,dc=com
```

```
orclVersion: 112000
orclcommonrpwdattribute: {SASL -MD5}eW5+2LTPRKzFmHxmMZQmnw==
objectClass: orclApplicationEntity
objectClass: orclService
objectClass: orclDBServer_92
objectClass; orclDBServer
objectClass: top
orclServiceType: DB
orclSid: orcl11g
oracleHome: /app/oracle/product/db/product/11.2.0/dbhome_1
cn: orcl11g
orclSystemName: oudhost
userPassord: {SSHA}oNeBEqkUMtDusjXNXJPpa7qa+Yd0b9RHvA==
orclNetDescString: (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST)=oudhost)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=orcl11g)))
orclDBGLOBALNAME: orcl11g
orclNetDescName: 000:cn=    DESCRIPTION_0
```

31.4.1.5 Configuring Roles and Permissions

The following topics provide the steps to configure roles and permissions using Oracle Enterprise Manager:

- Creating a Shared Schema in the Database
- Creating a New User-Schema Mapping
- Creating a Role in the Database
- Creating a New Role in the Domain
- Defining a Proxy Permission in the Database
- Creating a New Proxy Permission
- Configuring Mappings for a Specific Database

31.4.1.5.1 Creating a Shared Schema in the Database

Run the following SQL commands:

```
SQL> CREATE USER global_ident_schema_user IDENTIFIED GLOBALLY;
User created.
SQL> GRANT CONNECT TO global_ident_schema_user;
Grant succeeded.
```

31.4.1.5.2 Creating a New User-Schema Mapping

> **Note:**
>
> Before performing the steps mentioned in this procedure, see Configuring Password Policy for Oracle Unified Directory Administrator.

To create a new user schema mapping:

1. In a web browser, connect to Enterprise Manager. For example:

   ```
   https://localhost:1158/em
   ```

Provide the following, then click Login.

- **User Name**

  Enter the name of a user who is authorized to administer the database.

- **Password**

  Enter the administrator password.

- **Connect As**

  Choose SYSDBA.

  Click Login.

2. Click the Server tab.

   On the Server tab, in the Security section, click Enterprise User Security.

3. In the "Oracle Internet Directory Login: Enterprise User Security" page, provide the following information:

   - **User**

     Enter the username of a user, for example `cn=directory manager`, who has write access to Oracle Context.

   - **Password**

     Enter the password for the same user.

   Click Login.

4. On the Enterprise User Security page, click Manage Enterprise Domains.

   An Enterprise Domain can contain one or more databases. The settings for an Enterprise Domain apply to all databases it contains.

5. On the Manage Enterprise Domains page, select the domain you want to configure, then click Configure.

6. On the Configure Domain page, click "User - Schema Mappings."

7. On the User - Schema Mappings page, click Create.

8. To create a domain-schema mapping, on New Mapping page provide the following information:

   a. **From**

      You can associate a global schema to all the users in a given subtree, or to a given user.

      To associate a global schema to all users in a given subtree:

      1. Choose Subtree, then click the flashlight icon to search for available subtrees.

      2. In the Select User page, select a subtree.

      3. Enterprise users below the DN you select will be mapped to the same global schema. Click Select.

      To associate a global schema to a given user:

      1. Choose User Name, then click the flashlight icon to search for available users.

      2. In the select User page, select a user DN. Only this specific user will be mapped to the global schema. Click Select.

   b. **To**

1. In the Schema field, enter the name of the global schema.
2. For example, `global_ident_schema_user`.

Click Continue.

9. On the "User - Schema Mappings" tab, when you are satisfied that the mapping is correct, click OK.

31.4.1.5.3 Creating a Role in the Database

For this example, a role named `hr_access`, is created. The role grants read access to the table `hr.employees`.

To create a role in the database:

```
SQL> CREATE ROLE hr_access IDENTIFIED GLOBALLY;
Role created.
SQL> GRANT SELECT ON hr.employees TO hr_access;
Grant succeeded.
```

For more information, see the Oracle Database documentation.

31.4.1.5.4 Creating a New Role in the Domain

To create a new role in the domain:

1. On the Manage Enterprise Domains page, select the domain in which you want to create the role, then click Configure.
2. On the Configure Domain page, click Enterprise Roles. Click Create.
3. On the Create Enterprise Role page, provide the following information:
   a. In the Name field, provide a name for your enterprise role.
   b. In the DB Global Roles tab, click Add.
4. In the Search And Select: Database Global Roles page, provide the following information:
   - **Database**

     Choose the database from the drop-down list.
   - **User Name**

     Enterprise Manager will retrieve the available roles from the database. Enter a username of an administrator, for example `SYS AS SYSDBA`, who is authorized to access the roles.
   - **Password**

     Enter the administrator password.

   Click Go.
5. In the "Search and Select: Database Global Roles" page, choose the global role you want to grant to Enterprise Users.

   Click Select.
6. In the Create Enterprise Role page, select the Enterprise user or groups to which you will grant the Enterprise Role, then click the Grantees tab.
7. On the Grantees tab, to select Enterprise users or groups click Add.

8. In the "Select: Users and Groups" page, click Go. Enterprise Manager retrieves available Users and Groups.
   - **View**

     You can search for users or groups.
   - **Search Base**

     Enterprise Manager begins the search at this DN.
   - **Name**

     Enter a string here to narrow down the search. For example, if you want to find a user whose name starts with jo, enter **jo** and Click Go.

     A table displays relevant entries. From the list, select the users and groups to which you want to grant the Enterprise Role, then click Select.

     Click Continue.
9. In the Configure Domain page, click OK to continue.
10. In the Edit Enterprise Role page, click Continue.
11. In the Configure Domain page, click OK.

    After the role has been successfully created, click Configure.

31.4.1.5.5 Defining a Proxy Permission in the Database

To define a proxy permission on user SH, run the following command:

```
SQL> ALTER USER SH GRANT CONNECT THROUGH ENTERPRISE USERS;
User altered.
```

This command defines a proxy permission on user SH.

31.4.1.5.6 Creating a New Proxy Permission

To create a new proxy permission:

1. On the Configure Domain Information page, select the domain you want to configure, then click Configure.
2. On the Configure Domain page, click Proxy Permissions.
3. To create a new Proxy Permission, on the Proxy Permissions tab click Create.
4. On the Create Proxy Permission page, in the **Name** field, provide a name for your Proxy Permission.
5. On the Target DB Users tab, click Add.
6. On the "Search And Select: Database Target Users" page, provide the following information:
   - **Database**

     Choose the database from the drop-down list.
   - **User Name**

     Enter the username of an administrator, for example `SYS AS SYSDBA`, who is authorized to access the users.

- **Password**

  Enter the administrator password.

Click Go.

Enterprise Manager retrieves the available target users from the database.

7. In the Search and Select page, select the target user for the proxy permission, then click Select.
8. In the Create Proxy Permission page, click the Grantees tab.
9. On the Grantees tab, click Add.
10. On the Select Users and Groups page, click Go. Enterprise Manager retrieves available Enterprise Users.

    In the Select: Users and Groups page, select the users to be granted Proxy Permission. Then click Select to continue.
11. On the Create Proxy Permission page, click Continue.
12. On the Configure Domain page, click OK to continue.

31.4.1.5.7 Configuring Mappings for a Specific Database

To configure mappings for a specific database:

1. On the Enterprise User Security page, click Manage Databases.
2. On the Manage Databases page, select the database you want to configure, and click Configure.
3. On the Configure Database page, click "User - Schema Mappings" tab.
4. On the "User - Schema Mappings" page, click Create.
5. To create a domain-schema mapping, on New Mapping page provide the following information:

   a. **From**

      You can associate a global schema to all the users in a given subtree, or to a given user.

      To associate a global schema to all users in a given subtree:

      1. Choose Subtree, then click the flashlight icon to search for available subtrees.

      2. In the Select User page, select a subtree.

      3. Enterprise users below the DN you select will be mapped to the same global schema. Click Select.

      To associate a global schema to a given user:

      1. Choose User Name, then click the flashlight icon to search for available users.

      2. In the select User page, select a user DN. Only this specific user will be mapped to the global schema. Click Select.

   b. **To**

      1. In the Schema field, enter the name of the global schema.

      2. For example, `global_ident_schema_user`.

   Click Continue.

6. On the "User - Schema Mappings" tab, when you are satisfied that the mapping is correct, click OK.

31.4.1.6 Testing the Database Configurations

At this point Enterprise User Security contains the following configurations:

- A users-schema mapping granting a global schema to all users below `dc=example,dc=com`
- An Enterprise Role granting `HR_ACCESS` to `uid=user.0,ou=people,dc=example,dc=com`
- A Proxy Permission allowing `uid=user.1,our=people,dc=example,dc=com` to proxy user `SH`.

To test the database configurations:

1. Run `sqlplus` to connect to the database with `user.0`.

   In the following example, SQLPlus prompts for the user password. The administrator provides the password configured for `uid=user.0,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.0

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014

Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
CONNECT
HR_ACCESS

SQL>
```

   In this example, the following are indications that the database is configured properly for users such as `user.0`.

   - The line that starts with `Connected to:` indicates that authentication succeeded.
   - The line that begins with `SQL> select * from session_roles;` enables the user to check the roles granted to himself.
   - The database role `HR_ACCESS` is granted through the Enterprise Role.

2. Run `sqlplus` to connect to the database with `user.1` credentials.

   In the following example, SQLPlus prompts for the user password. The administrator provides the password configured for `uid=user.1,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.1

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014
```

```
Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
CONNECT


SQL>
```

In this example, the following are indications that the database is configured properly for users such as `user.1`.

- The line that starts with `Connected to:` indicates that authentication succeeded.
- The line that begins with `SQL> select * from session_roles;` enables the user to check the roles granted to himself.
- The only database role is `CONNECT`, and it is granted through the Global Schema.

3. Run `sqlplus` to connect to the database a with `user.1` credentials using a proxy permission as user `SH`.

   In the following example, SQLPlus prompts for the user password.The administrator provides the password configured for `uid=user.1,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.1[sh]

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014

Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
RESOURCE
SELECT_CATALOG_ROLE
HS_ADMIN_SELECT_ROLE
CWM_USER


SQL>
```

In this example, the following are indications that the database is configured properly for users such as `user.1`.

- The line that starts with `Connected to:` indicates that authentication succeeded.
- The line that begins with `SQL> select * from session_roles;` enables the user to check the roles granted to himself.
- The user `user.1` inherits the roles of user `SH` through the proxy authentication.

31.4.2 Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security

Follow these tasks to configure Oracle Unified Directory Proxy to work with an External LDAP Directory and Enterprise User Security.

Table 31-2 List of tasks to configure Oracle Unified Directory Proxy

Task #	Link
Task 1	Configuring User Identities in the External LDAP Directory
Task 2	Configuring Oracle Unified Directory Proxy to Work with Enterprise User Security
Task 3	Configuring the Users and Groups Location
Task 4	Selecting the Oracle Context to be Used By Enterprise User Security
Task 5	Registering the Database in the LDAP Server
Task 6	Configuring Roles and Permissions
Task 7	Testing the Database Configurations

31.4.2.1 Configuring User Identities in the External LDAP Directory

Configure the existing user and group identities so they can be recognized by Enterprise User Security. Choose from the following based on your external LDAP directory:

- Configuring User Identities in Microsoft Active Directory
- Configuring User Identities in Microsoft Active Directory Using Centrally Managed Users
- Configuring User Identities in Oracle Directory Server Enterprise Edition
- Configuring User Identities in Novell eDirectory
- Configuring User Identities in Oracle Unified Directory

31.4.2.1.1 Configuring User Identities in Microsoft Active Directory

In previous releases, you can integrate Oracle Database user's authentication and authorization with Active Directory by configuring Oracle Enterprise User Security, and installing and configuring Oracle Unified Directory. However, you can now authenticate and authorize Active Directory users with the database directly using Centrally Managed Users (CMU).
Starting with Oracle Unified Directory 12c (12.2.1.4.0), the Password Notification Change plug-in (`oidpwdcn.dll`) is deprecated. Oracle recommends that you use the CMU feature provided by Oracle Database. CMU supports all the newer and stronger hashing algorithms and other updated security enhancements.

If you are currently using the Password Notification Change plug-in and planning to transition to CMU, you must perform the following steps:

1. Remove `oidpwdcn.dll` from system32, if present.
2. Remove the entry `oidpwdcn` from the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\` registry.
3. Configure Centrally Managed Users with Microsoft Active Directory. See Configuring Centrally Managed Users with Microsoft Active Directory in *Oracle Database Security Guide*.

   See Also, Connecting to Microsoft Active Directory and Install the Password Filter and Extend the Microsoft Active Directory Schema for Password Authentication.

31.4.2.1.2 Configuring User Identities in Microsoft Active Directory Using Centrally Managed Users

In previous releases, you can integrate Oracle Database user's authentication and authorization with Active Directory by configuring Oracle Enterprise User Security, and installing and configuring Oracle Unified Directory. However, you can now authenticate and authorize Active Directory users with the database directly using Centrally Managed Users (CMU).
Starting with Oracle Unified Directory 12c (12.2.1.4.0), the Password Notification Change plug-in (`oidpwdcn.dll`) is deprecated. Oracle recommends that you use the CMU feature provided by Oracle Database. CMU supports all the newer and stronger hashing algorithms and other updated security enhancements.

If you are currently using the Password Notification Change plug-in and planning to transition to CMU, you must perform the following steps:

1. Remove `oidpwdcn.dll` from system32, if present.
2. Remove the entry `oidpwdcn` from the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\` registry.
3. Configure Centrally Managed Users with Microsoft Active Directory. See Configuring Centrally Managed Users with Microsoft Active Directory in *Oracle Database Security Guide*.

   See Also, Connecting to Microsoft Active Directory and Install the Password Filter and Extend the Microsoft Active Directory Schema for Password Authentication.

31.4.2.1.3 Configuring User Identities in Oracle Directory Server Enterprise Edition

Run `ldapmodify` command from Oracle Directory Server Enterprise Edition to enable extended operation for the account lock, as follows:

```
ldapmodify -h <ODSEE Server> -p <ODSEE port> -D <ODSEE Admin ID> -w <ODSEE Admin
password>
dn: oid=1.3.6.1.4.1.42.2.27.9.6.25,cn=features,cn=config
changetype: add
objectclass: directoryServerFeature
oid: 1.3.6.1.4.1.42.2.27.9.6.25
cn: Password Policy Account Management
```

31.4.2.1.4 Configuring User Identities in Novell eDirectory

Enable the Universal Password in eDirectory, and allow the administrator to retrieve the user password.

See the Novell eDirectory documentation about Password Management for more information.

31.4.2.1.5 Configuring User Identities in Oracle Unified Directory

Modify the default password policy to use Salted SHA-1 as password storage scheme by running `dsconfig` command as follows:

```
./dsconfig -h <OUD host> -p <OUD admin port> -D <OUD dirmgr> -j <pwdfile>
-X -n set-password-policy-prop\
--policy-name "Default Password Policy"\
--set default-password-storage-scheme:"Salted SHA-1"
```

You can configure the default password policy to use a more secure and a robust password storage scheme, namely EUS PBKDF2 SHA-512 if your Oracle RDBMS version supports it. Oracle recommends that you contact your Database Administrator to validate if the RDBMS version deployed at your end supports Multi-Round SHA-512 based password verifier.

> **Note:**
>
> Ensure that you modify the default password policy of Oracle Unified Directory containing the Enterprise Users and the Enterprise Groups details. Do not modify the default password policy of the Oracle Unified Directory instance acting as the proxy server.

31.4.2.2 Configuring Oracle Unified Directory Proxy to Work with Enterprise User Security

If you do not already have an Oracle Unified Directory Proxy installed, complete the steps in one of these sections:

- Installing and Configuring a New Oracle Unified Directory Proxy Using the Command Line.
- Installing and Configuring a New Oracle Unified Directory Proxy to Work with Enterprise User Security Using the Graphical User Interface.

If you already have an Oracle Unified Directory Proxy instance installed, complete the steps in Configuring an Existing Oracle Unified Directory Proxy to Work with Enterprise User Security Using OUDSM.

31.4.2.2.1 Installing and Configuring a New Oracle Unified Directory Proxy Using the Command Line

To install and configure the new Oracle Unified Directory Proxy:

1. Run the `oud-proxy-setup` command. For example:

   ```
   oud-proxy-setup -i -p 1389 --adminConnectorPort 4444
   -D "cn=directory manager" -j pwd.txt -Z 1636 --generateSelfSignedCertificate
   --eusContext dc=example,dc=com
   ```

2. Create an LDAP server extension for the remote LDAP server containing the Enterprise users and groups. For example:

   ```
   dsconfig create-extension \
            --set enabled:true \
            --set remote-ldap-server-address:serverip \
            --set remote-ldap-server-port:389 \
   ```

```
--type ldap-server \
--extension-name proxy1 \
--hostname localhost \
--port 4444 \
--trustAll \
--bindDN "cn=directory manager" \
--bindPasswordFile pwd.txt \
--no-prompt
```

3. Create a Proxy workflow element for the remote LDAP server using the LDAP server extension you created in the previous step.

   You can configure this Proxy workflow element to use either the `use-specific-identity` or the `use-client-identity` mode.

   - Use `use-specific-identity` mode if your external LDAP server does not allow anonymous access. This is the most common Enterprise User Security configuration, especially when Active Directory is used as the external LDAP server.

     To create the proxy workflow element using the `use-specific-identity` mode, run the `dsconfig` command as follows:

     ```
     dsconfig create-workflow-element \
              --set client-cred-mode:use-specific-identity \
              --set enabled:true \
              --set ldap-server-extension:proxy1 \
              --set remote-ldap-server-bind-dn: \
                cn=administrator,cn=users,dc=example,dc=com \
              --set remote-ldap-server-bind-password:******** \
              --set remote-root-dn:cn=administrator,cn=users,dc=example,dc=com\
              --set remote-root-password:******** \
              --type proxy-ldap \
              --element-name proxy-we1 \
              --hostname localhost \
              --port 4444 \
              --trustAll \
              --bindDN "cn=directory manager" \
              --bindPasswordFile pwd.txt \
              --no-prompt
     ```

     In this example, `remote-root-dn` and `remote-ldap-server-bind-dn` are the credentials used by OUD proxy to connect to the remote server.

   - Use `use-client-identity` mode if your external LDAP server allows anonymous access.

     If you want to use the `use-client-identity` mode, then you must configure the external LDAP server credentials and configure an exclude-list.

     The database usually connects with its own credentials to Oracle Unified Directory proxy server, and then performs searches on the external LDAP server. When EUS is enabled, the database must use an alternate ID to bind to the external LDAP server because the database entry does not exist on the external LDAP server. The database entry is stored locally on the Oracle Unified Directory proxy server.

     To create the proxy workflow element using `use-client-identity` mode, run the `dsconfig` command as follows:

     ```
     dsconfig create-workflow-element \
              --set client-cred-mode:use-client-identity \
              --set enabled:true \
              --set ldap-server-extension:proxy1 \
              --set exclude-list:"cn=directory manager" \
     ```

```
--set exclude-list:cn=oraclecontext,dc=example,dc=com \
--set remote-ldap-server-bind-dn: \
  cn=administrator,cn=users,dc=example,dc=com \
--set remote-ldap-server-bind-password:******** \
--set remote-root-dn:cn=administrator,cn=users,dc=example,dc=com\
--set remote-root-password:******** \
--type proxy-ldap \
--element-name proxy-we1 \
--hostname localhost \
--port 4444 \
--trustAll \
--bindDN "cn=directory manager" \
--bindPasswordFile pwd.txt \
--no-prompt
```

In this example, `remote-root-dn` and `remote-ldap-server-bind-dn` are the credentials used by the remote LDAP administrator.

Important. When in `use-client-identity` mode, if you are integrating with Active Directory, then you must also run the following command to allow anonymous login, where `dc=example,dc=com` is the base DN of your Active Directory server.

```
ldapmodify -h ADhost -p ADport -D ADdirmgr -w pwd
dn: cn=directory service,cn=windows
nt,cn=services,cn=configuration,dc=example,dc=com
changetype: modify
replace: dsHeuristics
dsHeuristics: 0000002
```

4. Create a EUS workflow element using the proxy workflow element created in the previous step:

```
dsconfig create-workflow-element \
          --set enabled:true \
          --set eus-realm:dc=example,dc=com \
          --set next-workflow-element:proxy-we1 \
          --set server-type:ad \
          --type eus \
          --element-name eus-we1 \
          --hostname localhost \
          --port 4444 \
          --trustAll \
          --bindDN "cn=directory manager" \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

Note: The `server-type` defines the remote LDAP server containing your enterprise users and groups. Use one of the following values: `ad` for Active Directory, `edir` for Novell eDirectory, `oud` for Oracle Unified Directory, or `odsee` Oracle Directory Server Enterprise Edition.

5. Create a workflow for your naming context using the EUS workflow element created in the previous step:

```
dsconfig create-workflow \
          --set base-dn:dc=example,dc=com \
          --set enabled:true \
          --set workflow-element:eus-we1 \
          --type generic \
          --workflow-name workflow1 \
          --hostname localhost \
          --port 4444 \
          --trustAll \
```

```
            --bindDN "cn=directory manager" \
            --bindPasswordFile pwd.txt \
            --no-prompt
```

6. Add the workflow created in the previous step to your network group:

```
dsconfig set-network-group-prop \
            --group-name network-group \
            --add workflow:workflow1 \
            --hostname localhost \
            --port 4444 \
            --trustAll \
            --bindDN "cn=directory manager" \
            --bindPasswordFile pwd.txt \
            --no-prompt
```

31.4.2.2.2 Installing and Configuring a New Oracle Unified Directory Proxy to Work with Enterprise User Security Using the Graphical User Interface

> **Note:**
>
> The OUD instance creation GUI wizard is deprecated in Oracle Unified Directory 12c (12.2.1.4.0). Oracle recommends use of the command-line (CLI) to create an instance. For more information, see Setting Up the Proxy Using the CLI.

To install and configure a new Oracle Unified Directory Proxy to work with Enterprise User Security using the graphical user interface:

1. Run the `oud-proxy-setup` program.
   a. In the Welcome page, click **Next**.
   b. In the Server Settings page, provide the following information:

      Host Name. Enter the name of the OUD proxy host.

      Administration Connector Port. This is the administration port used by OUD tools such as `dsconfig`.

      LDAP Listener Port. Specify the port used by the OUD proxy.

      LDAP Secure Access. Click **Configure** to enable secure access.

      In the Configure Secure Access window, click to mark the "Enable SSL on Port" check box. Then enter a port number for LDAPS, and click **OK** to continue.

      Root User DN. This is the identity of the server administrator.

      Password. Enter a password to be used by the server administrator.

      Password (confirm). Enter the password a second time to confirm.

      Click **Next** to continue.

   c. In the Deployment Options page, in the Configuration Option field, choose "Configure EUS (Enterprise User Security)" and click Next.

      Oracle Unified Directory will be used as a proxy, and deployed in front of the LDAP server containing EUS users and groups.

   d. On the Back-End Server Type page, choose one of the supported server types. This is the LDAP-compliant server that contains the Enterprise User Security users and groups.

Click **Next** to continue.

e. On the next page, click **Add Server**.

On the Add Server page, provide the following information:

Host Name. Enter the host name of the LDAP server that contains Enterprise User Security users and groups.

Protocol. If you are using Novell eDirectory, you must choose LDAPS.

For all other external directories, you can choose one of the following: LDAP, LDAPS, or [LDAP & LDAPS]. This determines how OUD proxy will connect to the remote LDAP server.

Port Number. Enter the port number of the LDAP server that contains Enterprise User Security users and groups.

You can click Add to add another LDAP server. After you are done adding LDAP servers, click **Close** to continue.

f. Review the list on the Servers Page.

The Servers Page now lists the server or servers that contain Enterprise User Security users and groups. Click **Next** to continue.

g. On the Naming Contexts page, click to mark the check box beside a Base DN to choose the Base DN for a naming context.

If the table does not display a Naming Context, enter the Base DN of your remote LDAP server in the "Additional Naming Context DN" field, select Add.

Click Next to continue.

h. Configure the runtime options for the server.

You can click Change to configure any specific JVM settings, or click Next to run the server with the default JVM settings.

Click Next.

i. In the Review page, review your settings, and click Finish.

A new instance of Oracle Unified Directory Proxy is installed, configured, and started.

Click Close.

2. Set the remote root DN and remote root user accounts by running the `dsconfig` command on the OUD Proxy as follows:

```
dsconfig set-workflow-element-prop \
          --element-name proxy-we1 \
          --set remote-root-dn:cn=directory manager \
          --set remote-root-password:******** \
          --hostname localhost \
          --port 4444 \
          --trustAll \
          --bindDN "cn=directory manager" \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

> **Note:**
>
> In the preceding command, `--element-name` property corresponds to the name of the proxy workflow element, which is used to connect to the external LDAP directory server.
>
> If you configure proxy through OUD proxy setup wizard, then the default name of the proxy workflow element is `proxy-we1`. Alternatively, if you configure the proxy through CLI by using `dsconfig` command, then the name of the workflow element would be as per the value you provide as an input in the command.
>
> You can find the workflow element by running the `dsconfig` command as follows:
>
> ```
> dsconfig -h localhost -p administration port number -D
> "cn=Directory Manager" -X -n list-workflow-elements --bindPasswordFile
> password.txt
> ```
>
> You observe output similar to the following:
>
> ```
> Workflow Element  : Type               : enabled
> ----------------- :--------------------:--------
> adminRoot         : ldif-local-backend : true
> load-bal-we1      : load-balancing     : true
> proxy-we1         : proxy-ldap         : true
> ```
>
> In the above example, if you look at the `proxy-ldap` type, you will locate the workflow element name (`proxy-we1`) corresponding to that.

3. Set the mode for the proxy workflow element for the external LDAP-compliant directory.

   By default, the configuration is set to `use-client-identity` mode.

   - Use `use-specific-identity` mode if your external LDAP server does not allow anonymous access. This is the most common Enterprise User Security configuration, especially when Active Directory is used as the external LDAP server.

     If you want to change the mode setting to `use-specific-identity`, then you must configure the external LDAP server credentials.

     To use `use-specific-identity` mode, run the `dsconfig` command as follows:

     ```
     dsconfig set-workflow-element-prop \
              --element-name proxy-we1 \
              --set client-cred-mode:use-specific-identity \
              --set remote-ldap-server-bind-dn: \
                cn=administrator,cn=users,dc=example,dc=com\
              --set remote-ldap-server-bind-password:******** \
              --hostname localhost \
              --port 4444 \
              --trustAll \
              --bindDN "cn=directory manager" \
              --bindPasswordFile pwd.txt \
              --no-prompt
     ```

     In this example, `remote-root-dn` and `remote-ldap-server-bind-dn` are the credentials used by the remote LDAP administrator.

   - Use `use-client-identity` mode if your external LDAP server allows anonymous access.

If you want to use the `use-client-identity` mode, then you must configure the external LDAP server credentials and an exclude-list.

The database usually connects with its own credentials to Oracle Unified Directory proxy server, and performs searches on the external LDAP server. When EUS is enabled, the database must use an alternate ID to bind to the external LDAP server because the database entry does not exist on the external LDAP server. The database entry is stored locally on the Oracle Unified Directory proxy server.

To use the `use-client-identity` mode, run the `dsconfig` command as follows:

```
dsconfig set-workflow-element-prop \
          --element-name proxy-we1 \
          --set client-cred-mode:use-client-identity \
          --add exclude-list:cn=directory manager \
          --add exclude-list:cn=oraclecontext,dc=example,dc=com \
          --set remote-ldap-server-bind-dn: \
            cn=administrator,cn=users,dc=example,dc=com \
          --set remote-ldap-server-bind-password:******** \
          --hostname localhost \
          --port 4444 \
          --trustAll \
          --bindDN "cn=directory manager" \
          --bindPasswordFile pwd.txt \
          --no-prompt
```

In this example, `remote-root-dn` and `remote-ldap-server-bind-dn` are the credentials used by the remote LDAP administrator.

Important. When in `use-client-identity` mode, if you are integrating with Active Directory, then you must run the following command to allow anonymous login, where `dc=example,dc=com` is the base DN of your Active Directory server.

```
ldapmodify -h <ADhost> -p <AD port> -D <AD dirmgr> -w <pwd>
dn: cn=directory service,cn=windows
nt,cn=services,cn=configuration,dc=example,dc=com
changetype: modify
replace: dsHeuristics
dsHeuristics: 0000002
```

31.4.2.2.3 Configuring an Existing Oracle Unified Directory Proxy to Work with Enterprise User Security Using OUDSM

To configure an existing Oracle Unified Directory Proxy to work with Enterprise User Security using OUDSM:

1. Connect to Oracle Unified Directory Proxy from OUDSM.
2. Select the Home tab.
3. Under the Configuration section, choose "Set Up Remote EUS Naming Context."
4. In the "Create Remote EUS Naming Context" page, provide the following information:

   Base DN. This is the suffix provided by the remote LDAP server.

   Network Group. Attach the suffix to at least one network group. Select the required network group from the list.

   Server Type. Select the type of LDAP server containing your users and groups from the list.

   Host Name. Enter the name of the machine where the remote LDAP server is running.

Ports available. Indicate whether you want the OUD Proxy to connect to the remote LDAP server using LDAP, or LDAPS, or both LDAP and LDAPS.

Depending upon the option you chose, enter a port number for the LDAP port, LDAPS port, or for both LDAP and LDAP ports. This must be the port used by the remote LDAP server.

If you checked LDAPS, configure SSL to either Trust All or configure a Trust Manager.

Click Create.

5. Select the Configuration tab.
6. In the Naming Contexts list, choose the Proxy below the Naming context you just created.
7. In the Proxy LDAP workflow element window:
   a. Enter a Bind DN and a Bind Password.

      These must match the credentials of the remote LDAP server administrator.

   b. Expand the Remote Root Properties, and enter a Remote Root DN and password.

      These must match the credentials of the remote LDAP server administrator.

   c. In the Credentials Mode field, set the mode for the proxy workflow element for the external LDAP-compliant directory.

      - Use `use-specific-identity` mode if your external LDAP server does not allow anonymous access. This is the most common Enterprise User Security configuration, especially when Active Directory is used as the external LDAP server.

        To use `use-specific-identity` mode:

        In the Credentials Mode field, choose Use Specific Identity. Then enter the values for the Bind DN and the Bind Password. Enter the Bind Password a second time to confirm it.

      - Use `use-client-identity` mode if your external LDAP server allows anonymous access.

        To `use-client-identity` mode:

        In the Credentials Mode field, first select Use Client Identity, and expand the Client Identity Mode Properties. Then add `"cn=directory manager"` and `"cn=OracleContext,dc=example,dc=com"` to the Exclude Bind DNs table.

   d. Click Apply.

31.4.2.3 Configuring the Users and Groups Location

After Oracle Unified Directory has been configured for EUS or Oracle E-Business Suite, you must configure the naming context used to store the users and the groups by performing the following steps:

1. Locate the LDIF template file at `install_dir/config/EUS/modifyRealm.ldif`.
2. Edit the `modifyRealm.ldif` file as follows:
   - Replace `dc=example,dc=com` with the correct naming context for your server instance.
   - Replace `ou=people` and `ou=groups` with the correct location of the user and group entries in your DIT.

3. Use the `ldapmodify` command to update the configuration with the edited LDIF template file, for example:

```
$ ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file -f
modifyRealm.ldif
```

> **Note:**
>
> Ensure that you specify the port number on which the LDAP Connection Handler will listen for connections from clients (For example, `1389`) and not the administration port number which is `4444`.

4. If you are integrating Active Directory, run the following command, replacing `dc=example,dc=com` with the appropriate base DN for your configuration:

```
$ ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file
dn:cn=Common,cn=Products,cn=OracleContext,dc=example,dc=com
changetype: modify
replace: orclCommonNickNameAttribute
orclCommonNickNameAttribute: samaccountname
```

31.4.2.4 Selecting the Oracle Context to be Used By Enterprise User Security

Enterprise User Security stores its configuration (also called EUS metadata) in an Oracle Context, which corresponds to a part of the Directory Information Tree. If your user entries are stored below `dc=example,dc=com`, then EUS is usually configured to use `cn=OracleContext,dc=example,dc=com` as Oracle Context.

In this task, Oracle Net Configuration Assistant tells EUS where it should read its configuration.

1. To start the Oracle Net Configuration Assistant, run the `netca` command on the host where the database is installed.

   The Oracle Net Configuration Assistant is displayed.

2. On the Welcome page, select "Directory Usage Configuration," and click **Next**.

   Enter the following information in subsequent pages:

   a. **Directory Type**

      Select "Oracle Internet Directory" even if the LDAP server is an Oracle Virtual Directory or an Oracle Unified Directory.

      Click **Next**.

   b. **Hostname**

      Enter the host name or IP address of the server hosting your LDAP server.

   c. **Port**

      Enter the LDAP port number.

   d. **SSL Port**

      Enter the LDAPS port number.

   e. **Oracle Context**

      Do not select `cn=OracleContext`. Instead, click the arrow to display and choose the location of your OracleContext.

Oracle Net Configuration Assistant connects to the LDAP server to retrieve the available Oracle Contexts. Enterprise User Security configuration will be stored within your `OracleContext`.

Click **Next**.

f. Directory usage configuration complete!

Click **Next**.

When the Welcome page is displayed, click **Finish**.

3. To verify that the Net Configuration Assistant has successfully created the configuration file containing the LDAP server information, run the following command:

```
# cat $ORACLE_HOME/network/admin/ldap.ora
# ldap.ora Network Configuration File: /app/oracle/product/db/product/11.2.0/
dbhome_1/network/admin/ldap.ora
# Generated by Oracle configuration tools.
DIRECTORY_SERVERS= (oudhost:1389:1636)
DEFAULT_ADMIN_CONTEXT = "dc=example,dc=com"
DIRECTORY_SERVER_TYPE = OID
```

The configuration file used by the database contains the host name and port of the LDAP server. In this example, the information is represented as: `(oudhost:1389:1636)`. You can specify multiple servers, separated by commas, for high availability deployments.

In this example, `dc=example,dc=com` represents the Oracle Context used to store the EUS configuration, also known as the EUS metadata.

31.4.2.5 Registering the Database in the LDAP Server

To register the database in the LDAP server:

1. Run the `dbca` command on the host where the database is installed.

The Database Configuration Assistant for Oracle database is displayed. Click Next, then provide the following information in the subsequent pages:

a. **Select the operation you want to perform.**

Choose "Configure Database Option," then click Next.

b. **Database**

In the list box, select the database you want to register. Then click Next.

Database Configuration Assistant determines if the database is already registered in the LDAP server.

c. **Would you like to register this database with the directory service?**

Choose "Yes, register the database." Database Configuration Assistant will create an entry for the database in the Oracle Context.

d. **User DN**

The user DN will be used to authenticate to the LDAP server.

The user DN is usually `cn=directory manager`, the directory manager of OUD proxy. The user DN is also used in the add operation, which creates the database entry in the Oracle Context. The user must have write access to the LDAP server.

e. **Password**

Database Configuration Assistant creates a wallet for the database. The database entry DN and password will be stored in the wallet. When the database connects to the LDAP server, it will authenticated using credentials stored in this wallet.

f. **Database Components**

Make no changes to this page, and click Next.

g. **Connection Mode**

Choose "Dedicated Server Mode," then click Finish.

h. **Confirmation**

Click OK to register the database.

i. **Do you want to perform another operation?**

Click No to exit the Database Configuration Assistant application.

2. To verify that Database Configuration Assistant successfully created a new entry for the database, run the following command, replacing `orcl11g` with the name of your database:

```
$ ldapsearch -h oudhost -p 1389 -D "cn=directory manager" -j pwd.txt -b
cn=oraclecontext,dc=example,dc=com "(cn=orcl11g)"
dn: cn=orcl11g,cn=OracleContext,dc=example,dc=com
orclVersion: 112000
orclcommonrpwdattribute: {SASL -MD5}eW5+2LTPRKzFmHxmMZQmnw==
objectClass: orclApplicationEntity
objectClass: orclService
objectClass: orclDBServer_92
objectClass; orclDBServer
objectClass: top
orclServiceType: DB
orclSid: orcl11g
oracleHome: /app/oracle/product/db/product/11.2.0/dbhome_1
cn: orcl11g
orclSystemName: oudhost
userPassord: {SSHA}oNeBEqkUMtDusjXNXJPpa7qa+Yd0b9RHvA==
orclNetDescString: (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST)=oudhost)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=orcl11g)))
orclDBGLOBALNAME: orcl11g
orclNetDescName: 000:cn=    DESCRIPTION_0
```

31.4.2.6 Configuring Roles and Permissions

The following topics provide the steps to configure roles and permissions using Oracle Enterprise Manager:

- Creating a Shared Schema in the Database
- Creating a New User-Schema Mapping
- Creating a Role in the Database
- Creating a New Role in the Domain
- Defining a Proxy Permission in the Database
- Creating a New Proxy Permission
- Configuring Mappings for a Specific Database

31.4.2.6.1 Creating a Shared Schema in the Database

Run the following SQL commands:

```
SQL> CREATE USER global_ident_schema_user IDENTIFIED GLOBALLY;
User created.
SQL> GRANT CONNECT TO global_ident_schema_user;
Grant succeeded.
```

31.4.2.6.2 Creating a New User-Schema Mapping

> **Note:**
>
> Before performing the steps mentioned in this procedure, see Configuring Password Policy for Oracle Unified Directory Administrator.

To create a new user schema mapping:

1. In a web browser, connect to Enterprise Manager. For example:

   `https://localhost:1158/em`

   Provide the following information:

   User Name. Enter the name of a user who is authorized to administer the database.

   Password. Enter the administrator password.

   Connect As. Choose SYSDBA.

   Click Login.

2. Click the Server tab.

   On the Server tab, in the Security section, click Enterprise User Security.

3. In the "Oracle Internet Directory Login: Enterprise User Security" page, provide the following information:

   User. Enter the username of a user, for example `cn=directory manager`, who has write access to Oracle Context.

   Password. Enter the password for the same user.

   Click Login.

4. On the Enterprise User Security page, click Manage Enterprise Domains.

   An Enterprise Domain can contain one or more databases. The settings for an Enterprise Domain apply to all databases it contains.

5. On the Manage Enterprise Domains page, select the domain you want to configure, then click Configure.

6. On the Configure Domain page, click "User - Schema Mappings."

7. On the User - Schema Mappings page, click Create.

8. To create a domain-schema mapping, on the New Mapping page provide the following information:

   From

   You can associate a global schema to all the users in a given subtree, or to a given user.

   To associate a global schema to all users in a given subtree**:**

   a. Choose Subtree, then click the flashlight icon to search for available subtrees.

b. In the Select User page, select a subtree. Enterprise users below the DN you select will be mapped to the same global schema.

c. Click Select.

To associate a global schema to a given user:

a. Choose User Name, then click the flashlight icon to search for available users.

b. In the select User page, select a user DN. Only this specific user will be mapped to the global schema.

c. Click Select.

To

In the Schema field, enter the name of the global schema. For example:`global_ident_schema_user`.

Click Continue.

9. On the "User - Schema Mappings" tab, when you are satisfied that the mapping is correct, click OK.

31.4.2.6.3 Creating a Role in the Database

For this example, a role named `hr_access`, is created. The role grants read access to the table `hr.employees`.

To create a role in the database:

```
SQL> CREATE ROLE hr_access IDENTIFIED GLOBALLY;
Role created.
SQL> GRANT SELECT ON hr.employees TO hr_access;
Grant succeeded.
```

For more information, see the Oracle Database documentation.

31.4.2.6.4 Creating a New Role in the Domain

To create a new role in the domain:

1. To create a new role in a domain, On the Manage Enterprise Domains page, select the domain in which you want to create the role, then click Configure.
2. On the Configure Domain page, click Enterprise Roles. Click Create.
3. On the Create Enterprise Role page, provide the following information:
   a. In the Name field, provide a name for your enterprise role.
   b. In the DB Global Roles tab, click Add.
4. On the "Search And Select: Database Global Roles' page, provide the following information:

   Database. Choose a database from the drop-down list.

   User Name. Enterprise Manager will retrieve the available roles from the database. Enter a username of an administrator, such as `SYS AS SYSDBA`, who is authorized to access the roles.

   Password. Enter the administrator password.

   Click Go.

5. In the "Search and Select: Database Global Roles" page, choose the global role you want to grant to Enterprise Users.

   Click Select.

6. In the Create Enterprise Role page, select the Enterprise user or groups to which you will grant the Enterprise Role, then click the Grantees tab.
7. On the Grantees tab, to select Enterprise users or groups click Add.
8. In the "Select: Users and Groups" page, click Go. Enterprise Manager retrieves available Users and Groups.

   View. You can search for users or groups.

   Search Base. Enterprise Manager begins the search at this DN.

   Name. Enter a string here to narrow down the search. For example, if you want to find a user whose name starts with jo, enter **jo** and Click Go.

   A table displays relevant entries. From the list, select the users and groups to which you want to grant the Enterprise Role, then click Select.

   Click Continue.

9. In the Configure Domain page, click OK to continue.
10. In the Edit Enterprise Role page, click Continue.
11. In the Configure Domain page, click OK.

    After the role has been successfully created, click Configure.

31.4.2.6.5 Defining a Proxy Permission in the Database

To define a proxy permission on user SH, run the following command:

```
SQL> ALTER USER SH GRANT CONNECT THROUGH ENTERPRISE USERS;
User altered.

This command defines a proxy permission on user SH.
```

31.4.2.6.6 Creating a New Proxy Permission

To create a new proxy permission:

1. On the Configure Domain Information page, select the domain you want to configure, then click Configure.
2. On the Configure Domain page, click Proxy Permissions.
3. To create a new Proxy Permission, on the Proxy Permissions tab click Create.
4. On the Create Proxy Permission page, in the **Name** field, provide a name for your Proxy Permission.
5. On the Target DB Users tab, click Add.
6. On the "Search And Select: Database Target Users" page, provide the following information:

   Database. Choose the database from the drop-down list.

   User Name. Enter the username of an administrator, for example `SYS AS SYSDBA`, who is authorized to access the users.

Password. Enter the administrator password.

Click Go.

Enterprise Manager retrieves the available target users from the database.

In the Search and Select page, select the target user for the proxy permission, then click Select.

7. In the Create Proxy Permission page, click the Grantees tab.
8. On the Grantees tab, click Add.
9. On the Select Users and Groups page, click Go. Enterprise Manager retrieves available Enterprise Users.

   In the Select: Users and Groups page, select the users to be granted Proxy Permission. Then click Select to continue.
10. On the Create Proxy Permission page, click Continue.
11. On the Configure Domain page, click OK to continue.

31.4.2.6.7 Configuring Mappings for a Specific Database

To configure mappings for a specific database:

1. On the Enterprise User Security page, click Manage Databases.
2. On the Manage Databases page, select the database you want to configure, and click Configure.
3. On the Configure Database page, click "User - Schema Mappings" tab.
4. On the "User - Schema Mappings" page, click Create.
5. To create a domain-schema mapping, on the New Mapping page provide the following information:

   From

   You can associate a global schema to all the users in a given subtree, or to a given user.

   To associate a global schema to all users in a given subtree:

   a. Choose Subtree, then click the flashlight icon to search for available subtrees.
   b. In the Select User page, select a subtree. Enterprise users below the DN you select will be mapped to the same global schema.
   c. Click Select.

   To associate a global schema to a given user:

   a. Choose User Name, then click the flashlight icon to search for available users.
   b. In the select User page, select a user DN. Only this specific user will be mapped to the global schema.
   c. Click Select.

   To

   In the Schema field, enter the name of the global schema. For example:`global_ident_schema_user`.

   Click Continue.

6. On the "User - Schema Mappings" tab, when you are satisfied that the mapping is correct, click OK.

31.4.2.7 Testing the Database Configurations

At this point Enterprise User Security contains the following configurations:

- A users-schema mapping granting a global schema to all users below `dc=example,dc=com`
- An Enterprise Role granting `HR_ACCESS` to `uid=user.0,ou=people,dc=example,dc=com`
- A Proxy Permission allowing `uid=user.1,our=people,dc=example,dc=com` to proxy user SH.

To test the database configurations:

1. Run `sqlplus` to connect to the database with `user.1` credentials using a proxy permission as user `SH`.

   In the following example, SQLPlus prompts for the user password. The administrator provides the password configured for `uid=user.0,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.0

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014

Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
CONNECT
HR_ACCESS

SQL>
```

   In this example, the following are indications that the database is configured properly for users such as user.0.

   - The line that starts with `Connect to:` indicates that authentication succeeded.
   - The line that begins with `SQL> select * from session_roles;` enables the administrator to check the roles granted to the Enterprise User.
   - The database role `HR_ACCESS` is granted through the Enterprise Role.

2. Run `sqlplus` to connect to the database as with `user.1` credentials using a proxy permission as user `SH`.

   In the following example, SQLPlus prompts for the user password. The administrator provides the password configured for `uid=user.1,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.1

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014

Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
CONNECT


SQL>
```

In this example, the following are indications that the database is configured properly for users such as user.1.

- The line that starts with `Connect to:` indicates that authentication succeeded.
- The line that begins with `SQL> select * from session_roles;` enables the administrator to check the roles granted to the Enterprise User.
- The only database role is `CONNECT`, and it is granted through the Global Schema.

3. Run `sqlplus` to connect to the database a with `user.1` credentials using a proxy permission as user `SH`.

   In the following example, SQLPlus prompts for the user password. The administrator provides the password configured for `uid=user.1,ou=people,dc=example,dc=com` in the LDAP server.

```
# sqlplus user.1[sh]

SQL*Plus: Release 11.2.0.2.0 Production on Fri Feb 7 16:16:04 2014

Copyright  (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select * from session_roles;


Role
------------------------------
RESOURCE
SELECT_CATALOG_ROLE
HS_ADMIN_SELECT_ROLE
CWM_USER


SQL>
```

In this example, the following are indications that the database is configured properly for users such as `user.1`.

- The line that starts with `Connect to:` indicates that authentication succeeded.
- The line that begins with `SQL> select * from session_roles;` enables the user currently logged in to check the roles granted to himself.
- The user `user.0` inherits user SH's roles through the proxy authentication.

31.4.3 Configuring Password Policy for Oracle Unified Directory Administrator

When you create the user-schema mapping you are required to provide the user name of the Oracle Unified Directory administrator, such as `cn=directory manager`, which is used to log in to Oracle Unified Directory server.

You must perform the following steps before creating the user-schema mapping:

1. Modify the password policy associated with the Oracle Unified Directory administrator to add `AES` as the default password storage scheme and to allow for multiple password values. For instance, if the administrator is `cn=directory manager` then modify the password policy as follows:

```
./dsconfig -h localhost -p port -D "cn=directory manager" -j pwdfile -X -n set-
password-policy-prop \
--policy-name "Root Password Policy" \
--add default-password-storage-scheme:AES

./dsconfig -h localhost -p port -D "cn=directory manager" -j pwdfile -X -n set-
password-policy-prop \
--policy-name "Root Password Policy" \
--set allow-multiple-password-values:true
```

2. Modify the LDAP password of the Oracle Unified Directory administrator as follows:

```
./ldappasswordmodify -X -Z -h localhost -p port -D "cn=directory manager" -j pwdfile
\
--currentPassword password --newPassword mynewpassword
```

31.5 Using Additional Enterprise User Security Configuration Options

After the basic integration of Oracle Unified Directory and Enterprise User Security, you can configure OUD to support multiple EUS domains and configure replication to support high availability.

- Configuring OUD to Support Multiple Enterprise User Security Domains.
- Using Oracle Unified Directory and Enterprise User Security in High Availability Topologies.

31.5.1 Configuring OUD to Support Multiple Enterprise User Security Domains

If your users and groups are stored in multiple domains, you must configure OUD to support multiple EUS domains. For example, a single OUD instance contains two EUS domains. One EUS domain stores users entries in Active Directory below `cn=users,dc=ad1,dc=com`. A

second EUS domain stores user entries in a different Active Directory instance below `cn=users,dc=ad2,dc=com`. You must configure OUD to support each EUS domain.

To configure OUD to support multiple EUS domains:

1. Configure OUD as if the primary domain is the single domain containing all your users and groups.

   In this example, the primary domain is `dc=ad1,dc=com`.

   Complete the tasks in Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security.

2. Configure the secondary domain.

   In this example, the secondary domain is `dc=ad2,dc=com`.

   For this secondary domain, complete the steps in Configuring User Identities in the External LDAP Directory.

3. Create a new naming context for the EUS domain, which is `dc=ad2,dc=com` in this example.

   Complete the steps in Configuring an Existing Oracle Unified Directory Proxy to Work with Enterprise User Security Using OUDSM.

4. Update the Oracle context with the new naming context.

   a. Create an LDIF file.

      In the following `myconfig.ldif` example, make the following substitutions:

      - Replace `dc=ad1,dc=com` with the DN of your first domain.
      - Replace `orclcommonusersearchbase` with the users location in the secondary domain.

        ```
        dn: cn=Common,cn=Products,cn=OracleContext,dc=ad1,dc=com
        changetype: modify
        add: orclcommonusersearchbase
        orclcommonusersearchbase: cn=users,dc=ad2,dc=com
        ```

      - Replace `orclcommongroupsearchbase` with the groups location in the secondary domain.

        ```
        dn: cn=Common,cn=Products,cn=OracleContext,dc=ad1,dc=com
        changetype: modify
        add: orclcommongroupsearchbase
        orclcommongroupsearchbase: cn=groups,dc=ad2,dc=com
        ```

   b. Update OUD configuration using the LDIF file you created in step 4a.

      ```
      ldapmodify -h oudhost -p 1389 -D "cn=directory manager" -w password -f
      myconfig.ldif
      ```

31.5.2 Using Oracle Unified Directory and Enterprise User Security in High Availability Topologies

You can achieve high availability among two or more OUD instances that have been integrated with Enterprise User Security. First, integrate OUD with Enterprise User Security. Then configure replication among the integrated OUD instances. Once configured, replication takes

place among Enterprise User Security metadata (in either directory server or directory proxy) and the OUD server users and groups.

Configuring an integrated OUD LDAP server for replication is the same as configuring an integrated OUD Proxy server *with one exception*: the list of suffixes to be replicated is different.

When an integrated OUD instance is configured as an LDAP server, the following suffixes are replicated:

```
cn=oraclecontext
cn=oraclecontext,dc=example,dc=com
dc=example,dc=com
```

When an integrated OUD instance is configured as a Proxy server, the following suffixes are replicated:

```
cn=oraclecontext
cn=oraclecontext,dc=example,dc=com
```

> **Note:**
>
> If you are using Oracle Data Guard or Oracle Real Application Clusters or high availability, each database instance must be configured using NetCA and DBCA.

To configure OUD-EUS integrated instances for high availability:

1. Enable the first Oracle Unified Directory and Oracle Enterprise User Security to work together.
   - If the first OUD instance is a directory server, then complete the tasks in Configuring Oracle Directory Server as a Directory for Enterprise User Security.
   - If the first OUD instance is a directory proxy, then complete the tasks in Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security.
2. Enable the second Oracle Unified Directory instance and Oracle Enterprise User Security to work together.
   - If the second OUD instance is configured as an LDAP server, then complete the tasks in Configuring Oracle Directory Server as a Directory for Enterprise User Security.
   - If the second OUD instance is configured as a proxy, then complete the tasks in Configuring Oracle Unified Directory Proxy to Work with an External LDAP Directory and Enterprise User Security.
3. Enable replication between the first OUD instance and the second OUD instance.
   - If the OUD instance is an LDAP server, then run this command:

```
# dsreplication enable --host1 oud-proxy-source --port1 4444 --bindDN1
 "cn=Directory Manager"  --bindPasswordFile1 /tmp/pwd1.txt
 --replicationPort1 repl1 --host2 oud-proxy-dest --port2 4444 --bindDN2
 "cn=Directory Manager"  --bindPasswordFile2 /tmp/pwd2.txt
 --replicationPort2 repl2 --adminUID admin --adminPasswordFile
 /tmp/pwd3.txt --baseDN "cn=OracleContext,dc=example,dc=com" --baseDN
 "cn=OracleContext" --baseDN "dc=example,dc=com" -X -n
```

- If the OUD instance is a directory proxy, then run this command:

```
# dsreplication enable --host1 oud-proxy-source --port1 4444 --bindDN1
 "cn=Directory Manager"  --bindPasswordFile1 /
tmp/pwd1.txt --replicationPort1 repl1 --host2 oud-proxy-dest --port2 4444
 --bindDN2 "cn=Directory Manager"  --bindPasswordFile2 /tmp/pwd2.txt
 --replicationPort2 repl2 --adminUID admin --adminPasswordFile
 /tmp/pwd3.txt --baseDN "cn=OracleContext,dc=example,dc=com" --baseDN
 "cn=OracleContext" -X -n
```

> **Note:**
>
> In the directory proxy example, the `--baseDN "dc=example,dc=com"` option is not included.

Replication is now enabled in the first OUD instance (from step 1), and in the second OUD instance (from step 2).

4. Initialize replication. For example:

- If the OUD instance is a directory server, then run this command:

```
dsreplication initialize  --baseDN "cn=OracleContext,dc=example,dc=com"
  --baseDN "cn=OracleContext" --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPasswordFile /tmp/pwd3.txt \
  --hostSource <oud-proxy-source> --portSource 4444 \
  --hostDestination <oud-proxy-dest>  --portDestination 4444 -X -n
```

- If the OUD instance is a directory proxy, then run this command:

```
dsreplication initialize  --baseDN "cn=OracleContext,dc=example,dc=com" \
  --baseDN "cn=OracleContext" \
  --adminUID admin --adminPasswordFile /tmp/pwd3.txt \
  --hostSource <oud-proxy-source> --portSource 4444 \
  --hostDestination <oud-proxy-dest>  --portDestination 4444 -X -n
```

> **Note:**
>
> In the directory proxy example, the `--baseDN "dc=example,dc=com"` option is not included.

Both OUD instances now contain the same data. For more information, see Initializing a Replicated Server With Data.

5. Declare both OUD instances in the Database `ldap.ora` configuration file.

```
# ldap.ora Network Configuration File: /app/oracle/product/db/product/11.2.0/
dbhome_1/network/admin/ldap.ora
# Generated by Oracle configuration tools.
DIRECTORY_SERVERS= (oudhost1:1389:1636,oudhost2:1389:1636)
DEFAULT_ADMIN_CONTEXT = "dc=example,dc=com"
DIRECTORY_SERVER_TYPE = OID
```

31.6 Best Practices for Employing EUS Admin User

Enterprise User Security (EUS) requires a privileged user who can make changes to database information within the directory and reset users' passwords. Although the root DN user

(cn=Directory Manager for OUD/ODSEE or cn=orcladmin for OID/OVD) of a directory could do this, the best practice should be to use a least privileged user to administer EUS. This user is referred as the EUS Admin User (cn=eusadmin).

See:

- Overview of EUS Admin User
- Updating EUS Realm to Grant Administrative Privileges to EUS Admin Users
- Creating and Applying Password Policy for EUS Admin Users

31.6.1 Overview of EUS Admin User

Enterprise User Security (EUS) requires a privileged user who can make changes to database information within the directory and reset users' passwords. Although the root DN user (cn=Directory Manager for OUD/ODSEE or cn=orcladmin for OID/OVD) of a directory could do this, the best practice is to use a least privileged user to administer EUS. This user is referred as the EUS Admin User (cn=eusadmin).

The location of this user in the DIT structure is also important. If you place the EUS Admin User in any branch beyond the "cn=OracleContext, <Suffix>" branch in OUD, then the EUS Admin User will not be able to change their password because of constraints in the EUScontext workflow element. You should place the EUS Admin User in a local backend, so that it can be granted the password-reset privilege. If the implementation stores users and groups within a local backend, you can store the EUS Admin User in that backend. However, if the implementation proxies the users and groups through to a separate backend directory service, you should not store the user there. There is one other local backend suffix in the OUD instance that will exist for all OUD EUS deployments. That is the cn=OracleContext local backend. Although not required, it might be the best to place the EUS Admin users under cn=OracleContext for consistency.

Here is a sample EUS admin user:

```
dn: cn=eusadmin,ou=EUSAdmins,cn=OracleContext
objectClass: top
objectClass: organizationalperson
objectClass: inetorgperson
uid: cn=eusadmin,ou=EUSAdmins,cn=OracleContext
cn: eusadmin
sn: EUS
givenName: Admin
userPassword: password
ds-privilege-name: password-reset
ds-privilege-name: unindexed-search
```

31.6.2 Updating EUS Realm to Grant Administrative Privileges to EUS Admin Users

You must update the EUS Realm to grant administrative privileges to the EUS Admin Users. You can do this by making them a member of the respective EUS Administrative Groups.

The following example shows how to add the new EUS admin user cn=eusadmin,ou=EUSAdmins,cn=OracleContext to OracleContextAdmins groups. You can use this example as a reference to create the LDIF file to add the new user to all the required groups as per the Enterprise User Security Administrator's Guide.

See *Administrative Groups* in the *Enterprise User Security Administrator's Guide* to know about the respective EUS Administrative Groups.

To add the EUS Admin Users to the respective groups perform the following:

```
dn: cn=OracleContextAdmins,cn=Groups,cn=OracleContext
changetype: modify
add: uniqueMember
uniqueMember: cn=eusadmin,ou=EUSAdmins,cn=OracleContext

dn: cn=OracleContextAdmins,cn=groups,cn=OracleContext,dc=example,dc=com
changetype: modify
add: uniqueMember
uniqueMember: cn=eusadmin,ou=EUSAdmins,cn=OracleContext
```

31.6.3 Creating and Applying Password Policy for EUS Admin Users

If you run Oracle Database 14c with EUS, you would need the EUS Admin User to support the SASL DIGEST-MD5 authentication scheme. This requires that the uid value of the user be set to the full DN of that user and that the password policy for the EUS Admin Users include a reversible encryption storage scheme for the users' password such as AES, Base64, Blowfish, Clear, RC4 or TripleDES. So, you must create a password policy for the EUS Admin Users and apply that policy to the EUS Admin Users.

Procedure:

1. Create a password policy for the EUS Admin Users.

```
dsconfig create-password-policy \
          --set password-attribute:userpassword \
          --set default-password-storage-scheme:AES \
          --set default-password-storage-scheme:Salted\ SHA-512 \
          --type generic \
          --policy-name EUSAdmins \
          --hostname host.example.com \
          --port 1444 \
          --trustAll \
          --bindDN cn=Directory\ Manager \
          --bindPasswordFile /oracle/kit/.pw \
          --no-prompt
```

2. Apply this password policy to the EUS Admin User with ldapmodify using the following LDIF:

```
dn: cn=eusadmin,ou=EUSAdmins,cn=OracleContext
changetype: modify
add: ds-pwp-password-policy-dn
ds-pwp-password-policy-dn: cn=EUSAdmins,cn=Password Policies,cn=config
```

31.7 Understanding Enterprise User Security Password Warnings

Password policies are a set of rules that apply to all user passwords in an identity management realm. Password policies include settings for password complexity, minimum password length, and so forth. They also include account lockout and password expiration settings.

The database communicates with Oracle Unified Directory and requests the Oracle Unified Directory to report any password policy violations. If the database gets a policy violation

response from Oracle Unified Directory, then it displays the appropriate warning or error message to the user. The following table summarizes password warnings and their meanings.

Table 31-3 Password Warnings

Warning Condition	Message Example

The user password is about to expire. Message indicates the number of days left for the user to change his or her password.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28055: the password will expire within 1 days`
	
	`Connected.`

The password has expired and informs the user about the number of grace logins that remain.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28054: the password has expired. 1 Grace logins are left`
	
	`Connected.`

The user password has expired and the user does not have any grace logins left.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28049: the password has expired`

The user account has been locked due to repeated failed attempts at login.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28051: the account is locked`

The user account has been disabled by the administrator.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28052: the account is disabled`

The user account is inactive.	`SQL> connect joe/Admin123`
	`ERROR:`
	`ORA-28053: the account is inactive`

Enterprise user login attempts to the database update the user account status in Oracle Unified Directory or any supported external LDAP-compliant directory. For example, consecutive failed login attempts to the database results in the account getting locked in the directory, as per the directory's password policy.

31.8 Troubleshooting Issues after Integrating OUD and Enterprise User Security

You may encounter problems after integrating OUD and Enterprise User Security and need information on how to troubleshoot those problems.

These topics suggest solutions to issues you may encounter after integrating OUD and Enterprise User Security:

- Resolving Net Configuration Assistant Tool Error Messages
- Resolving Database Configuration Assistant Error Messages

- Resolving Oracle SQL Error Messages

31.8.1 Resolving Net Configuration Assistant Tool Error Messages

Find out how to resolve error messages reported by the Net Configuration Assistant (NetCA) Tool while integrating OUD and Enterprise User Security.

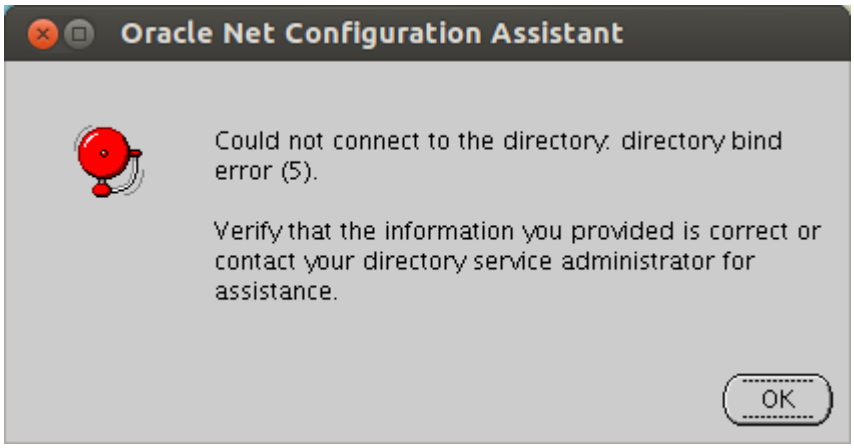
The following topics describe the Net Configuration Assistant (NetCA) Tool error messages and solutions:

- Resolving LDAP Server Connection Error
- Resolving Schema Error
- Resolving Naming Context Error

31.8.1.1 Resolving LDAP Server Connection Error

If the NetCA fails to connect to the directory then the Oracle Net Configuration Assistant screen displays the following error message:

Figure 31-1 Connection Error



To resolve this error, verify that the host name and port number are correct by running the following command on the command line:

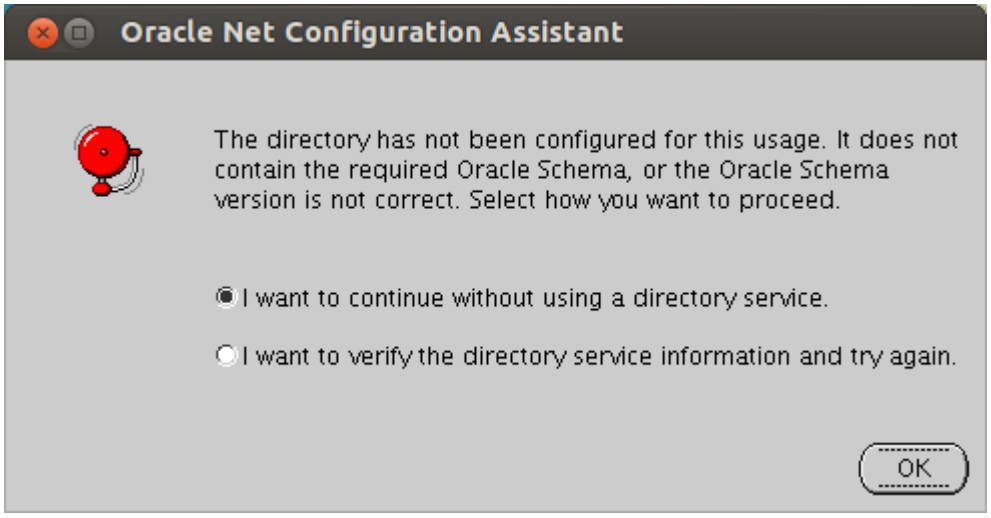
```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT  -b "" -s base
"(objectclass=*)"
dn:
objectClass: top
objectClass: ds-root-dse

$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $LDAPSPORT -Z -X  -b "" -s
base "(objectclass=*)"
dn:
objectClass: top
objectClass: ds-root-dse
```

31.8.1.2 Resolving Schema Error

If the required schema is not available or the version number is incorrect then the Oracle Net Configuration Assistant screen displays the following error message:

Figure 31-2 Oracle Schema



To resolve this error, ensure that you can access Oracle Unified Directory anonymously and that it contains the `cn=subschemasubentry` entry:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $LDAPSPORT -Z -X  -b
cn=subschemasubentry -s base "(objectclass=*)"
dn: cn=subschemasubentry
objectClass: top
objectClass: ldapSubentry
objectClass: subschema
```

If the Oracle Unified Directory is not enabled for Enterprise User Security then the `cn=subschemasubentry` entry will not be available. To enable Enterprise User Security, see "Setting up the Directory Server by Using the GUI" in the *Installing Oracle Unified Directory*.

If the `cn=subschemasubentry` is not accessible anonymously then ensure that the following ACI is defined in the Oracle Unified Directory as a global ACIs:

```
(target="ldap:///cn=subschemasubentry")(targetscope="base") \
(targetattr="objectClass||attributeTypes||dITContentRules||dITStructureRules| \
|ldapSyntaxes||matchingRules||matchingRuleUse||nameForms||objectClasses") \
(version 3.0; acl "User-Visible SubSchemaSubentry Operational Attributes"; \
allow (read,search,compare) userdn="ldap:///anyone";)
```

For more information, see Managing Global ACIs Using `dsconfig`.

31.8.1.3 Resolving Naming Context Error

If the `cn=OracleContext` and `cn=OracleContext,<your baseDN>` naming contexts are not available, then the Oracle Net Configuration Assistant screen displays an error message.

To resolve this error:

1. Verify if the baseDN is available, by running the following command on the command line:

   ```
   $ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $LDAPSPORT -Z -X  -b "" -s
   base "(objectclass=*)" namingContexts
   dn:
   namingContexts: cn=OracleContext
   ```

```
namingContexts: cn=OracleSchemaVersion
namingContexts: dc=eusovd,dc=com
```

As shown above, ensure that there are three available naming contexts. If the base DN is missing then you must enable Enterprise User Security, as described in "Setting up the Directory Server by Using the GUI" in the *Installing Oracle Unified Directory*.

2. Verify if the baseDN contains the Oracle context by running the following command on the command line:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $LDAPSPORT -Z -X  -b ""
"(objectclass=orclcontext)"
dn: cn=OracleContext
orclVersion: 90600
cn: OracleContext
objectClass: orclContext
objectClass: orclContextAux82
objectClass: top
objectClass: orclRootContext

dn: cn=OracleContext,dc=eusovd,dc=com
orclVersion: 90600
cn: OracleContext
objectClass: orclContext
objectClass: orclContextAux82
objectClass: top
```

> **Note:**
>
> The NetCA performs the search anonymously. If the Oracle Unified Directory is configured to refuse anonymous searches or the ACIs restricts access to `cn=OracleContext,<baseDN>` then the NetCA will not be able to find the Oracle Context.

3. After the NetCA configuration is complete, it creates an `ldap.ora` file in the `$ORACLE_HOME/network/admin` directory (UNIX) or `ORACLE_HOME\network\admin directory` (Windows). Ensure that it includes the following parameters:

```
DIRECTORY_SERVERS= (oudhost:1389:1636)
DEFAULT_ADMIN_CONTEXT = "dc=eusovd,dc=com"
DIRECTORY_SERVER_TYPE = OID
```

31.8.2 Resolving Database Configuration Assistant Error Messages

Find out how to resolve error messages reported by the Database Configuration Assistant (DBCA) while integrating OUD and Enterprise User Security.

The following topics describe the Database Configuration Assistant (DBCA) error messages and solutions:

- Resolving TNS-04409 error / TNS-04427: SSL access to the Directory Server
- Resolving TNS-04409 error / TNS-04431: Required suffixes
- Resolving TNS-04411 error when registering the DB with a user different from cn=directory manager
- Resolving TNS-04409 error / TNS-04405

31.8.2.1 Resolving TNS-04409 error / TNS-04427: SSL access to the Directory Server

This error message appears if SSL is not enabled for Oracle Unified Directory.

To resolve this error, check if SSL is enabled for Oracle Unified Directory by running the following command on the command line:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $LDAPSPORT -Z -X  -b "" -s
base "(objectclass=*)"
dn:
objectClass: top
objectClass: ds-root-dse
```

For more information, see Configuring Security Between Clients and Servers

31.8.2.2 Resolving TNS-04409 error / TNS-04431: Required suffixes

This error message appears if the suffixes are not available.

To resolve this error, ensure that the suffixes are created, as described in "Setting up the Directory Server by Using the GUI" in the *Installing Oracle Unified Directory*.

31.8.2.3 Resolving TNS-04411 error when registering the DB with a user different from cn=directory manager

This error message appears if you specify a different user name other then `cn=directory manager` during database registration.

To resolve this error, ensure that the user has password reset privilege, and the user entry contains one of the following `uniqueMember` attributes:

- `cn=oraclecontextadmins,cn=groups,cn=oraclecontext,dc=eusovd,dc=com`
- `cn=oraclenetadmins,dc=oraclecontext,dc=eusovd,dc=com`

  Run the following command on the command line:

```
$ OracleUnifiedDirectory/bin/ldapmodify -h $LDAPSERVER -p $LDAPPORT -D $DN -w $PWD
dn: cn=newadmin,ou=people,dc=eusovd,dc=com
changetype: modify
add: ds-privilege-name
ds-privilege-name: password-reset

Processing MODIFY request for cn=newadmin,ou=people,dc=eusovd,dc=com
MODIFY operation successful for DN cn=newadmin,ou=people,dc=eusovd,dc=com
dn: cn=oraclenetadmins,cn=oraclecontext,dc=eusovd,dc=com
changetype: modify
add: uniquemember
uniquemember:  cn=newadmin,ou=people,dc=eusovd,dc=com

Processing MODIFY request for cn=oraclenetadmins,cn=oraclecontext,dc=eusovd,dc=com
MODIFY operation successful for DN
cn=oraclenetadmins,cn=oraclecontext,dc=eusovd,dc=com
dn: cn=oraclecontextadmins,cn=groups,cn=oraclecontext,dc=eusovd,dc=com
changetype: modify
add: uniquemember
uniquemember:  cn=newadmin,ou=people,dc=eusovd,dc=com
```

```
Processing MODIFY request for
cn=oraclecontextadmins,cn=groups,cn=oraclecontext,dc=eusovd,dc=com
MODIFY operation successful for DN
cn=oraclecontextadmins,cn=groups,cn=oraclecontext,dc=eusovd,dc=com
```

31.8.2.4 Resolving TNS-04409 error / TNS-04405

This error message appears if the Oracle Unified Directory password validator does not accept the password that DBCA creates for the database entry (For example, if it requires a password minimum length of 10 characters).

To resolve this error:

1. Disable the password validator by running the following command on the command line:

```
$ OracleUnifiedDirectory/bin/dsconfig -h $LDAPSERVER -p $ADMINPORT \
-D $DN -j pwd.txt set-password-policy-prop \
--policy-name Default\ Password\ Policy --reset password-validator \
--trustAll --no-prompt
```

2. Run the `dbca` command.
3. Enable the password validator by running the following command on the command line:

```
$ OracleUnifiedDirectory/bin/dsconfig -h $LDAPSERVER -p $ADMINPORT -D
 $DN -j pwd.txt set-password-policy-prop --policy-name Default\
Password\ Policy --set password-validator:Length-Based\ Password\ Validator --
trustAll --no-prompt
```

31.8.3 Resolving Oracle SQL Error Messages

Find out how to resolve error messages reported by Oracle SQL while integrating OUD and Enterprise User Security.

The following topics describe the Oracle SQL error messages and solutions:

- Resolving ORA-28030: Server encountered problems accessing LDAP directory service
- Resolving ORA-01017: invalid username/password; logon denied
- Resolving ORA-28274: No ORACLE password attribute corresponding to user nickname exists
- Resolving ORA-28051: the account is locked

31.8.3.1 Resolving ORA-28030: Server encountered problems accessing LDAP directory service

This error message appears, if there is a problem with the connection between the database and the directory.

To resolve this issue:

1. Check that the database wallet has auto-login enabled. Either use Oracle Wallet Manager or check that there is a `cwallet.sso` file in `$ORACLE_HOME/admin/<ORACLE_SID>/wallet/`.
2. Check the DN and password of the user entry by running the following commands:

```
$ mkstore -wrl $ORACLE_BASE/admin/$ORACLE_SID/wallet -viewEntry ORACLE.SECURITY.DN
Oracle Secret Store Tool : Version 11.2.0.2.0 - Production
Copyright (c) 2004, 2010, Oracle and/or its affiliates. All rights reserved.
Enter wallet password:   ********
```

```
ORACLE.SECURITY.DN = cn=orcl11gr2,cn=OracleContext,dc=eusovd,dc=com

$ mkstore -wrl $ORACLE_BASE/admin/$ORACLE_SID/wallet -viewEntry
ORACLE.SECURITY.PASSWORD
Oracle Secret Store Tool : Version 11.2.0.2.0 - Production
Copyright (c) 2004, 2010, Oracle and/or its affiliates. All rights reserved.
Enter wallet password:   ********
ORACLE.SECURITY.PASSWORD = zQ7v4ek3
```

3. Check that the database can connect to the directory server using the following command:

```
$ oracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT
-b cn=common,cn=products,cn=oraclecontext,$BASEDN  "(objectclass=*)"
orclcommonusersearchbase orclcommongroupsearchbase orclcommonnicknameattribute
orclcommonnamingattribute
dn: cn=Common,cn=Products,cn=OracleContext,dc=eusovd,dc=com
orclcommonusersearchbase: ou=people,dc=eusovd,dc=com
orclcommongroupsearchbase: ou=groups,dc=eusovd,dc=com
orclcommonnicknameattribute: uid
orclcommonnamingattribute: cn
```

If the connection to the directory server fails, then you must do the following:

a. Ensure that the database entry exists in the Directory Server.

b. Ensure that the database entry contains a password in the `orclcommonrpwdattribute`, by running the following command:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT
-b  cn=oraclecontext,$BASEDN -s one "(objectclass=orcldbserver)"
 orclcommonrpwdattribute
dn: cn=orcl11gr2,cn=OracleContext,dc=eusovd,dc=com
orclcommonrpwdattribute: {SASL-MD5}KvIVAyYahxnHWdlfN649Kw==
```

If the entry is missing or does not contain a password then you must use DBCA, as described in Registering the Database in the LDAP Server.

31.8.3.2 Resolving ORA-01017: invalid username/password; logon denied

This error message appears, if an invalid username or password is provided.

To resolve this error, specify the correct username and password.

1. Check the Enterprise User Security configuration by running the following command:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT -b \
cn=common,cn=products,cn=oraclecontext,$BASEDN \
"(objectclass=*)" orclcommonusersearchbase \
orclcommongroupsearchbase orclcommonnicknameattribute orclcommonnamingattribute
dn: cn=Common,cn=Products,cn=OracleContext,dc=eusovd,dc=com
orclcommonusersearchbase: ou=people,dc=eusovd,dc=com
orclcommongroupsearchbase: ou=groups,dc=eusovd,dc=com
orclcommonnicknameattribute: uid
orclcommonnamingattribute: cn
```

After Oracle Unified Directory has been configured for EUS, the users and groups configurations are stored in the attributes `orclcommonusersearchbase` and `orclusercommongroupsearchbase`.

The username provided to sqlplus must correspond to the value of `orclcommonnicknameattribute` in the user entry. For example, if you connect sqlplus using the values joe/password and `orclcommonnicknameattribute=uid`, then the database will look for an entry containing the attribute `uid=joe`.

The user entry DN must start with `orclcommonnamingattribute`. For example, if `orclcommonnamingattribute=cn`, the user entry must be `cn=joe,<orclcommonusersearchbase>`.

2. Ensure that there is a user entry in the user container that matches the username provided in sqlplus. The inetorgperson objectclass, containing the attribute defined in orclcommonnicknameattribute.

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT \
-D $DN -w $PWD -b ou=people,$BASEDN  "(ui \d=joe)"
dn: cn=joe,ou=people,dc=eusovd,dc=com
userPassword: {SSHA}DdW5je5GCUnT2jVTeMdfPR9NWwkBt40FwWImpA==
objectclass: person
objectclass: organizationalPerson
objectclass: inetorgperson
objectclass: top
uid: joe
cn: joe
sn: joe
```

3. Ensure that you have created the user-schema mapping, as described in "Mapping Enterprise Users to the Shared Schema" in the *Oracle Database Enterprise User Security Administrator's Guide*.

31.8.3.3 Resolving ORA-28274: No ORACLE password attribute corresponding to user nickname exists

This error message appears, when the database finds a corresponding user but cannot compare its password with the password supplied to SQL.

To resolve this issue:

1. Ensure that the database entry has the required ACI to read the entry `authpassword` and `orclguid`:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT -D $DN
-w $PWD -b ou=people,$BASEDN  "(uid=joe)" authpassword orclguid
dn: cn=joe,ou=people,dc=eusovd,dc=com
authpassword;orclcommonpwd: {SSHA}DdW5je5GCUnT2jVTeMdfPR9NWwkBt40FwWImpA==
orclguid: 6458c6945c0a48be92ab35cf71859210
```

2. If the database cannot read the entry, check that the following ACIs are defined in your OUD server as global-acis (they are added automatically by oud-setup when EUS is selected):

```
(target="ldap:///dc=eusovd,dc=com")(targetattr!="userpassword||authpassword
||aci")(version 3.0; acl "Anonymous read access to subtree";allow
 (read,search,compare) userdn="ldap:///anyone";)
(target="ldap:///dc=eusovd,dc=com")(targetattr="authpassword||userpassword")
(version 3.0; acl "EUS reads authpassword"; allow (read,search,compare)
userdn="ldap:///??sub?(&(objectclass=orclservice)(objectclass=orcldbserver))";)
```

3. If the user entry does not contain authpassword, ensure that there is a user password:

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT -D $DN -w $PWD -b
ou=people,$BASEDN  "(uid=joe)" userpassword
dn: cn=joe,ou=people,dc=eusovd,dc=com
userpassword: {SSHA}DdW5je5GCUnT2jVTeMdfPR9NWwkBt40FwWImpA==
```

4. Ensure that the `userpassword` attribute is stored using a compatible scheme (SSHA-512 is not supported):

```
$ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT -D $DN -w $PWD -b
ou=people,$BASEDN  "(uid=joe)" userpassword
dn: cn=joe,ou=people,dc=eusovd,dc=com
userpassword: {SSHA}DdW5je5GCUnT2jVTeMdfPR9NWwkBt40FwWImpA==
```

31.8.3.4 Resolving ORA-28051: the account is locked

This error message appears, if you fail to authenticate properly after multiple attempts.

To resolve this issue:

1. Verify if Oracle Unified Directory is configured for account lockout, by running the following command on the command line:

   ```
   $ OracleUnifiedDirectory/bin/ldapsearch -h $LDAPSERVER -p $PORT -X -Z -D $DN
   -w $PWD -b "cn=Default Password Policy,cn=Password Policies,cn=config"
   "(objectclass=*)" ds-cfg-lockout-failure-count ds-cfg-lockout-duration  ds-cfg-
   lockout-failure-expiration-interval
   dn: cn=Default Password Policy,cn=Password Policies,cn=config
   ds-cfg-lockout-failure-expiration-interval: 180 s
   ds-cfg-lockout-failure-count: 3
   ds-cfg-lockout-duration: 180 s
   ```

   If the `failure-count` value is 0, then the account lockout is not enabled. For more information, see Managing Password Policies.

2. Ensure that the following ACI is defined, when the Enterprise User Security is configured:

   ```
   (target="ldap:///dc=eusovd,dc=com")(targetattr="orclaccountstatusevent")
   (version 3.0; acl "EUS write orclaccountstatusenabled"; allow (write)
   userdn="ldap:///??sub?(&(objectclass=orclservice)(objectclass=orcldbserver))";)
   (targetcontrol="2.16.840.1.113894.1.8.16")(version 3.0; acl "Anonymous control
   access"; allow(read) userdn="ldap:///anyone";)
   (targetcontrol="2.16.840.1.113894.1.8.2")(version 3.0; acl "Anonymous control
    access"; allow(read) userdn="ldap:///anyone";)
   ```

31.9 Disabling the Existing Anonymous ACIs in Upgraded Environments

When Oracle Unified Directory is used as the directory for Enterprise User Security, before 12.2.1.3.0, anonymous ACI was granted for EUS integration. In such upgraded environments, the existing anonymous global-aci in Oracle Unified Directory can be modified as follows to restrict anonymous search requests to Oracle Unified Directory.

`"dc=oracle,dc=com"` in the example should be replaced with the actual deployment specific DN.

```
(target="ldap:///dc=oracle,dc=com")(targetattr!="userpassword||authpassword||aci")
(targetfilter="(objectclass=orclContext)")(version 3.0; acl "Anonymous read access to
subtree";allow (read,search,compare) userdn="ldap:///anyone";)
(target="ldap:///dc=oracle,dc=com")(targetattr="*")(version 3.0; acl "EUS reads
authpassword"; allow (read,search,compare) userdn="ldap:///??sub?
(&(objectclass=orclservice)(objectclass=orcldbserver))";)
```

When Oracle Unified Directory Proxy is used to work with an External LDAP Directory and Enterprise User Security, the following virtual-acis can be added to restrict anonymous search requests to Oracle Unified Directory Proxy. `"dc=oracle,dc=com"` in the example should be replaced with the actual deployment specific DN.

```
(target="ldap:///dc=oracle,dc=com")(targetattr="*")(version 3.0; acl "EUS reads users";
allow (read,search,compare) userdn="ldap:///??sub?(&(objectclass=orclservice)
(objectclass=orcldbserver))";)
(target="ldap:///dc=oracle,dc=com")(targetattr="orclaccountstatusevent")(version 3.0;
acl "EUS write orclaccountstatusenabled"; allow (write) userdn="ldap:///??sub?
(&(objectclass=orclservice)(objectclass=orcldbserver))";)
(target="ldap:///dc=oracle,dc=com")(targetattr="*")(version 3.0; acl "Proxy self entry
access"; allow (read,search,compare,write) userdn="ldap:///self";)
```

Part VI

Advanced Administration: Data Replication, Schema Management, and Moving Across Environments

You can configure, monitor and troubleshoot data replication, manage the schema, and move server instances from a test environment to a production environment.

Topics:

32
Replicating Directory Data

Oracle Unified Directory supports replication to enable copies of identical data to be available across multiple servers.
Topics:

> **Note:**
>
> For information about the mechanics of the replication process, see Understanding the Oracle Unified Directory Replication Model.

32.1 About the Prerequisites Before Configuring Replication

Make sure you review the prerequisites before attempting to configure replication.

Note the following issues:

- Determine whether the default multi-master replication model is right for your deployment.

  The multi-master replication model is *loosely consistent* by default. This means that changes made on one server are replayed asynchronously to the other servers in the topology. The same entries can be modified simultaneously on different servers. When updates are sent between the two servers, any conflicting changes must be resolved.

Various attributes of a WAN, such as latency, can increase the chance of replication conflicts. Conflict resolution generally occurs automatically. Several conflict rules determine which change takes precedence. In some cases, conflicts must be resolved manually.

> **Note:**
>
> In certain deployment scenarios, the default loose consistency model might not be adequate. In these situations, you can configure replication to function in *assured* mode. For more information, see Configuring Assured Replication.

- SSL must be enabled. Replication always occurs over a secure connection. Both parties of a replication session must authenticate to the other using SSL certificates. No access control or privileges are enforced.
- You can set up replication automatically using the graphical setup utility when you first install Oracle Unified Directory only if you configure all of the directory servers in the same manner.
- You cannot use the `setup` command to configure replication in command-line mode. If you set up your directory servers by using the `setup` command, you must use the `dsreplication` command to configure replication between the servers.
- In any topology, you should have two replication servers for availability, in case one replication server fails. Replication servers are responsible for keeping track of all changes in the environment. Each replication server contains a list of all other replication servers in the topology.

> **Note:**
>
> In a replication architecture, each replication server is connected to every other replication server in the topology.

- To use the Dynamic Host Configuration Protocol (DHCP) in a replicated topology, you cannot change the replication servers' host names after the initial configuration.
- The examples in this section assume that you have already installed two directory servers and populated one with data. The directory servers can be installed on the same host machine, but if they are, they must have different port numbers.

32.2 Understanding Data Replication With `dsreplication`

The `dsreplication` command accesses the server configuration over SSL through the administration connector.

For more information, see Managing Administration Traffic to the Server.

The topics in this section include:

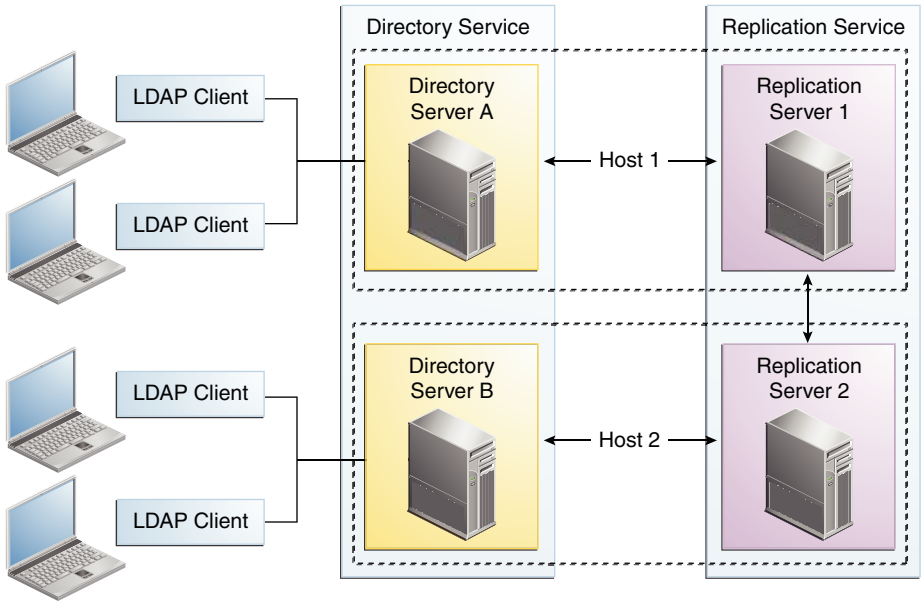
- Understanding Replication Between Two Servers With `dsreplication`
- Initializing a Replicated Server With `dsreplication`
- Initializing an Entire Topology With `dsreplication`
- Testing the Replicated Topology
- Obtaining the Status of a Replicated Topology With `dsreplication`

- Merging Two Existing Replicated Topologies With `dsreplication`
- Disabling Replication for a Specific Replication Domain With `dsreplication`
- Disabling Replication for a Specific Replication Domain With `dsreplication`

32.2.1 Understanding Replication Between Two Servers With `dsreplication`

You cannot run more than one instance of the `dsreplication enable` command to set up replication among multiple directory servers. You must run the `dsreplication enable` command separately for each directory/replication server pair in the topology.

Figure 32-1 Basic Replication Architecture



This section includes the following topics:

- Enabling Replication Between Two Servers With `dsreplication`
- Controlling Where Replication Servers are Created

32.2.1.1 Enabling Replication Between Two Servers With `dsreplication`

To enable replication between two directory servers:

Run the `dsreplication enable` command.

The following configuration example enables replication of the data under `"dc=example,dc=com"` between two directory servers: Directory Server A on `host1` and Directory Server B on `host2`. Both directory servers use the default administration port (`4444`). The command creates a replication server instance on `host1`, port `8989`, and a second replication server instance on `host2`, port `8989`.

```
$ dsreplication enable
  --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPasswordFile1 pwd.txt --replicationPort1 8989 \
```

```
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 pwd.txt --replicationPort2 8989 \
  --adminUID admin --adminPasswordFile pwd.txt --baseDN "dc=example,dc=com" -X -n
```

The `--adminUID` and `--adminPasswordFile` options refer to the Global Administrator for the replication domain. For more information, see Managing Administrators. The `-X` option specifies that all server certificates should be trusted and the `-n` (`--no-prompt`) option specifies that the command should be run in non-interactive mode. For information about all the global options for the `dsreplication` command, type `dsreplication -help` at the command-line.

If the host is a cluster node with logical IPs configured. You can configure the replication-listen-address to have multiple replication servers in a single cluster node listening on different logical IP's.

By default the replication server listen on 0.0.0.0:<replication port>, which will not allow you to have multiple replication server on a single cluster node with the same replication port.

The following configuration example enables replication of the data under "dc=example,dc=com" between two directory servers: Directory Server A on host1 - logical IP1 and Directory Server B on host1 - logical IP2. Both directory servers use the default administration port (4444). The command creates a replication server instance on host1 - logical IP1, port 8989, and a second replication server instance on host1 - logical IP2, port 8989.

```
       $ dsreplication enable
       --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
       --bindPasswordFile1 pwd.txt --listenAddress1 logical-ip1 --
replicationPort1 8989
      \
      --host2 host1 --port2 4444 --bindDN2 "cn=Directory Manager" \
      --bindPasswordFile2 pwd.txt --listenAddress2 logical-ip2 --
replicationPort2 8989
     \
      --adminUID admin --adminPasswordFile pwd.txt --baseDN
"dc=example,dc=com" -X -n
```

If a host has several network interfaces configured (not including loopback addresses), then you can specify them when providing the values for `--host1` and `--host2`. Use commas to separate values. For example:

```
$ dsreplication enable
  --host1 interface1,interface2,interface3 --port1 4444 --bindDN1 \
    "cn=Directory Manager" \
  --bindPasswordFile1 pwd.txt --replicationPort1 8989 \
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 pwd.txt --replicationPort2 8989 \
```

To add a new directory server to the replication topology, run `dsreplication enable` providing the connection information of the new server and the information of any of the already replicated servers.

To add replicas to an existing replication topology, for each replica that you want to add, run the following command:

```
$ dsreplication enable
  --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPasswordFile1 pwd.txt --replicationPort1 8989 \
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 pwd.txt --replicationPort2 8989 \
  --adminUID admin --adminPasswordFile pwd.txt --baseDN "dc=example,dc=com" -X -n
```

where `--[parameter]1` specifies an existing replica that has already been added to the replication topology, and `--[parameter]2` specifies the new replica to be added.

32.2.1.2 Controlling Where Replication Servers are Created

Using `dsreplication enable` between two servers automatically configures a replication server on each host. You might want to configure replication between two directory servers without creating a replication server on each host. Use the `--noReplicationServer1` or `--noReplicationServer2` options to add a directory server to a topology without creating an additional replication server. Remember that a replicated topology must contain at least two replication servers to avoid a single point of failure.

You can also enable replication between two servers and specify that one of the servers should only contain a replication server (not a directory server). Use the `--onlyReplicationServer1` or `--onlyReplicationServer2` options to achieve this. Specifying this option will configure a change log and replication port on the server the server will not contain replicated data.

32.2.2 Initializing a Replicated Server With `dsreplication`

Use the `dsreplication initialize` command to initialize a replicated server with the data from another replicated server.

The following command initializes the base DN `"dc=example,dc=com"` on host2 with the data contained on host1:

```
$ dsreplication initialize --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPasswordFile pwd.txt \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 -X -n
```

32.2.3 Initializing an Entire Topology With `dsreplication`

If there are more than two directory servers in the topology, use the `dsreplication intialize-all` command to initialize all replicas simultaneously. This command takes the details of the source host as arguments, and initializes all other servers for which replication is enabled.

The following command initializes all servers on which replication is enabled, from the contents of the base DN `"dc=example,dc=com"` on host1:

```
$ dsreplication initialize-all --hostname host1 --port 4444 \
  --baseDN "dc=example,dc=com" --adminUID admin --adminPasswordFile pwd.txt
```

32.2.4 Testing the Replicated Topology

The easiest way to test that replication is working is to apply changes on one directory server and to check that those changes have been replicated on another directory server.

To test the replication topology set up in the previous procedures:

1. Use `ldapmodify` to change an entry on host1.
2. Use `ldapsearch` to verify that the change was propagated to host2.

32.2.5 Obtaining the Status of a Replicated Topology With `dsreplication`

You can use the connection details of any directory server in the topology to obtain the status of the entire topology. Use the `dsreplication status` command to display a list of the directory servers in the topology, along with any missing changes between those servers.

The following command displays the status of the topology set up in the previous procedures:

```
$ dsreplication status --adminUID admin --adminPasswordFile pwd.txt -X \
  --hostname host1 --port 4444
```

32.2.6 Merging Two Existing Replicated Topologies With `dsreplication`

You can merge two replicated topologies by enabling replication between one server of each topology.

1. To merge two replicated topologies, use the `dsreplication enable` command.

   For example, if you have a replicated topology (for example, topology A) that includes host1, host2 and host3 and a replicated topology (for example, topology B) that includes host4, host5, and host6, the following command effectively merges the two topologies:

   ```
   $ dsreplication enable \
     --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
     --bindPasswordFile1 pwd.txt --replicationPort1 8989 \
     --host2 host4 --port2 4444 --bindDN2 "cn=Directory Manager" \
     --bindPasswordFile2 pwd.txt --replicationPort2 8989 \
     --adminUID admin --adminPasswordFile pwd.txt --baseDN "dc=example,dc=com" \
     -X -n
   ```

   This example assumes that both the hosts (host1 and host4) include a directory server and a replication server. If they do not, a directory server or replication server is automatically configured.

2. To ensure high availability, you must perform the following steps on all servers that were offline or unavailable during a merge:

   a. Initialize the contents of the suffix `cn=admin data` by using `dsreplication enable`

      You can initialize the servers individually, using one of the servers that was available during the merge, or you can use `dsreplication initialize-all`.

   b. Use the `dsconfig` command to update the list of replication servers.

Note the following limitations on merging two existing replicated topologies:

- All of the servers in both topologies must be up and running when you perform the merge.

  If a server its offline, `dsreplication` cannot update its configuration. If a server is offline when a merge is done, that server will not include the references to the replication servers in the other topology when it comes back online.

- The merge cannot be performed if there are conflicting domain IDs or replication server IDs between the two topologies.

  That is, a server in topology A cannot have the same replication server ID or domain ID as a server in topology B.

If there are conflicting IDs, the ID of the first server (`--host1`) is used to resolve the conflict. You must then re-initialize any servers that are out of date, using a server from the same topology as `--host1` as the source.

- Both replication topologies must have the same global administrators defined.

32.2.7 Disabling Replication for a Specific Replication Domain With `dsreplication`

You can disable replication for specific replication domain using the `dsreplication disable` command.

1. To disable replication on a specific domain, use the `dsreplication disable` command.

   The following command disables replication of the data under `"dc=example,dc=com"`.

   ```
   $ dsreplication disable --hostname host1 --port 4444 --adminUID admin \
     --adminPasswordFile pwd.txt --baseDN "dc=example,dc=com" -X -n
   ```

   This command removes the replication configuration from the directory server for that domain. If the domain that is disabled is the only replicated domain on this directory server instance, the command also disables the replication server on that instance. If the replication server is disabled, other directory servers that were connected to that replication server are disconnected and automatically reconnect to another replication server in the topology.

2. To disable the replication server itself (including the change log and the replication port) use the following command:

   ```
   $ dsreplication disable --hostname host1 --port 4444 -X -n \
     --adminUID admin --adminPasswordFile pwd.txt --baseDN "dc=example,dc=com" \
     --disableReplicationServer
   ```

   When the replication server is disabled, other directory servers that were connected to that replication server are disconnected and automatically reconnect to another replication server in the topology.

Notes about disabling the Replication Server

Disabling a replication server deletes the replication configuration but does *not* delete the replication server databases. You can therefore retrieve replication changes in the event that the replication server was disabled in error. If you have no requirement for re-enabling replication on this suffix, remove the replication server databases manually, for example: `$rm changelogDB/*`.

If replication is disabled, and then reenabled, any changes made on that server in the interim are not replicated. You must therefore either forbid changes on the server on which replication is disabled (for the period that replication is disabled) or resynchronize the rest of the topology from that server in the event that changes have occurred.

32.3 Configuring Data Replication Using OUDSM

Most server configuration that can be done by using Oracle Unified Directory Services Manager (OUDSM) is done from the Directory Manager tab. However, you can use either the Directory Manager tab or the Topology Manager tab to manage replication configuration.

- To view or configure replication configuration properties that are specific to an individual server or replicated suffix, select the Directory Manager tab.

- To manage an existing topology or to create a brand new topology using a replication configuration wizard, select the Topology Manager tab.

The topics in this section include:

- Considerations When Updating OUDSM.
- Viewing or Modifying an Existing Replication Server Configuration
- Viewing or Modifying a Replicated Suffix Configuration
- About Replication Configuration Wizard on the Directory Manager Tab.
- Accessing Replication Configuration Wizard from the Topology Manager Tab.

32.3.1 Considerations When Updating OUDSM

Make sure you consider these points while preparing to update the Oracle Unified Directory Services Manager (OUDSM), if you are using multiple instances of OUDSM in your replication topology.

Note the following:

- If you update one OUDSM instance, you must update all OUDSM and replicated instances.
- When updating OUDSM, you must also update Oracle Unified Directory to the same version. Updated OUDSM versions are not guaranteed to work with older Oracle Unified Directory versions.

> **Note:**
>
> For information about updating Oracle Unified Directory and Oracle Unified Directory Services Manager, see "Updating Oracle Unified Directory Services Manager on Oracle WebLogic Server" in the *Installing Oracle Unified Directory*.

32.3.2 Viewing or Modifying an Existing Replication Server Configuration

Select the Directory Manager tab in Oracle Unified Directory Services Manager (OUDSM) to view or modify the configuration of an existing replication server.

To view or modify an existing replication server:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Click the **Directory Manager** tab.
3. Click the tab of the server your want to configure.
4. Click the **Configuration** subtab.
5. In the Naming Contexts section, under General Configuration, click **Replication Server**.

   The Replication Server page is displayed.
6. View or modify the Replication Server properties.

   For a description of all possible properties, and their values, see the "Replication Server" section in the *Configuration Reference for Oracle Unified Directory.*

7. Click **Apply** to save any modifications you may have made.

32.3.3 Viewing or Modifying a Replicated Suffix Configuration

Select the Directory Manager tab in Oracle Unified Directory Services Manager to view or modify the configuration of a replicated suffix.

To view or modify a replicated suffix's configuration:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Click the **Directory Manager** tab.
3. Click the tab of the server your want to configure.
4. In the Naming Contexts section, under General Configuration, expand the **Replicated Suffixes** node, then select the suffix you want to view or modify.
5. Click the **Main** subtab.

   View or modify the properties on the Main subtab, then click **Apply** to save any modifications you may have made.

   For a description of all possible properties, and their values, see the "Replication Domain" section in the *Configuration Reference for Oracle Unified Directory.*
6. Click the **Assured Replication** subtab.

   View or modify the properties on the Assured Replication subtab, then click **Apply** to save any modifications you may have made.

   For a description of all possible properties, and their values, see the "Replication Domain" section in the *Configuration Reference for Oracle Unified Directory.*
7. Click the **Fractional Replication** subtab.

   View or modify the properties on the Fractional Replication subtab, then click **Apply** to save any modifications you may have made.

   For a description of all possible properties, and their values, see the "Replication Domain" section in the *Configuration Reference for Oracle Unified Directory.*

32.3.4 About Replication Configuration Wizard on the Directory Manager Tab

Launch the replication configuration wizard when you want to create a brand new topology, or to add a server to an existing topology.

You can launch a replication configuration wizard from the Directory Manager Tab when either of these conditions are true:

- No replication topology exists. See Creating a New Topology from Scratch.
- A replication topology exists, but the current server has not yet been added to it, or the current server is only partially configured for replication. See Adding a Server to an Existing Topology.

32.3.4.1 Creating a New Topology from Scratch

To create a new topology from scratch:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Click the **Directory Manager** tab.
3. Click the tab of the server your want to configure.
4. In the Naming Context pane, under General Configuration, choose **Replication Configuration**.

   The Replication Configuration page is displayed.
5. To launch the replication configuration wizard, click **Configure**.
6. In the Replication Options page, choose "**Do you want to create a new topology from scratch?**"

   Click **Next**.
7. In the Identify Servers page, enter the following information for at least two *source servers* to be configured:
   - **Host.** Enter the server name using the fully qualified domain name.
   - **Administration Port.** Enter the administration port number. The default is 4444.
   - **Admin Username.** Enter the DN for the administrator who can manage the server.
   - **Admin Password.** Enter the password for the administrator you specified.
8. (Optional) On this page, you can also perform the following:
   - To preview the suffixes configured for a server, click its **Preview Suffixes** link in the last column.
   - To add another server to be configured, click **Add**, then repeat step 6 above.
   - To remove a server from the topology, select the server name, then click **Remove**.

     Click **Next**.
9. In the Global Administrators page, the Domain Administrator can manage multiple directory server instances using OUDSM. This administrator is the Global Administrator who will manage the new replication topology.

   Provide the following information for the Global Administrator:
   - **Global Admin User ID.** This is the administrator who can view and manage the topology.
   - **Global Admin Password.** Enter the password for the Global Administrator specified above.
   - **Confirm Global Admin Password.** Enter the password again to confirm it.

   Click **Next**.
10. In the Replication Servers page, in the Configure Replication Servers table, the following information for each *replication server* is displayed:
    - **Host.** The replication server host name cannot be modified here.
    - **Admin Port.** The replication server admin port cannot be modified here.
    - **Act as Replication Server**. If you want the server to act as a replication server, then click the checkbooks until a check is displayed. If you cannot modify this setting, then the server is already configured as a replication server.
    - **Replication Port**. If you enabled a server to act as a replication server in the previous field, then enter a replication port number.

> **Note:**
>
> Be sure to enter a replication port number that is not already in use. If you cannot modify this setting, then the server is already configured as a replication server.

11. In the Replication Data page, the Configure Replicated Data table displays all the suffixes that are available in at least two servers among all servers. Indicate whether each suffix in the topology will be replicated. The suffixes you enable here will be replicated on all the servers in the replication topology.
    - To enable a domain to act as a replication suffix, in the Configure Replicated section, select a domain from the "Available for Replication" column, then click the right arrow to move it to the "Selected for Replication" column.
    - To enable a server to act as a replication domain, click its Replicate Suffixes checkbox until a checkmark is displayed.

    Click **Next**.
12. The summary page displays the replication server and domain information you just entered.
    - If you must modify any of the displayed information, click **Back**.
    - When you are satisfied that the Summary information is correct, click **Create**.

32.3.4.2 Adding a Server to an Existing Topology

To add a server to an existing topology:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Click the **Directory Manager** tab.
3. Click the tab of the server your want to configure.
4. In the Naming Context pane, under General Configuration, choose **Replication Configuration**.

   The Replication Configuration page is displayed.
5. To launch the replication configuration wizard, click **Configure**.
   - If the current server is already partially configured for replication, then it already exists as part of an existing topology. Skip step 6, and proceed to step 7.
   - If the current server is not already part of an existing topology, then the Replication Options page is displayed. Proceed to step 6.
6. In the Replication Options page, choose "**Is there an existing topology you want to add the server to?**"

   Click **Next**.
7. On the Connect/Identity Server page, in the Connect to Server section, the following information about the server you want to connect to in the existing topology is displayed:
   - **Host.** If the current server host is already part of a topology, then its name cannot be modified here. If the server is not part of a topology, then enter the host name of an existing server in the topology.

- **Administration Port.** If the current server is already part of a topology, then its administration port cannot be modified here. If the current server is not part of topology, then enter the administration port for the host specified above.
- **Global Admin User ID.** Enter the Global admin User ID. This is the administrator who can view and manage the topology. The user ID was specified when the topology was created.
- **Global Admin Password.** Enter the password for the Global Administrator specified above.

Click **Connect**. The "List of Servers" and "List of Replicated Suffixes" are displayed.

8. Review the "List of Servers" and the "List of Replicated Suffixes" to be sure that you are adding the server to the appropriate topology.

   When you are satisfied with the information displayed, click **Next**.

9. In the Replication Servers page, in the Configure Replication Servers table, the following information for each *replication server* to be configured is displayed:

   - **Host.** The replication server host name cannot be modified here.
   - **Admin Port**. The replication server administration port cannot be modified here.
   - **Act as Replication Server.** If you want the server to act as a replication server, then click the checkbox until a check is displayed. If you cannot modify this setting, then the server is already configured as a replication server.
   - **Replication Port.** If you enabled a server to act as a replication server in the previous field, then enter its replication port number.

   > **Note:**
   >
   > Be sure to enter a replication port number that is not already in use. If you cannot modify this setting, then the server is already configured as a replication server.

   Click **Next**.

10. In the Replication Data page, the Configure Replicated Data table displays all servers that contain suffixes already configured for replication in the topology, and that you've chosen to be added to the topology. Indicate whether each server in the topology will have suffixes replicated.

    To enable a server to act as a replication domain, click its Replicate Suffixes checkbox until a checkmark is displayed.

    Click **Next**.

11. The summary page displays the replication server and domain information you just entered.

    - If you must modify any of the displayed information, click **Back**.
    - When you are satisfied that the Summary information is correct, click **Apply**.

32.3.5 Accessing Replication Configuration Wizard from the Topology Manager Tab

Launch the replication configuration wizard when you want to create a brand new topology, or to add a server to an existing replication topology.

You can launch a replication configuration wizard from the topology Manager Tab when either of these conditions are true:

- No replication topology exists. See Creating a New Topology from Scratch.
- A replication topology exists, but the current server has not been added to the topology. See Managing an Existing Replication Topology.

32.3.5.1 Creating a New Topology from Scratch

To create a new topology from scratch:

1. To invoke OUDSM, enter the following URL into your browser's address field:

   `http://`*host*`:`*port*`/oudsm`

   where *host* is the name of the host on which OUDSM is running, and *port* is the port number of its administration server. The default administration port number is 7001.
2. Click the Topology Manager tab.

   The Topology Connections tab is displayed.
3. In the Create Replication Topology section of the Topology Connection tab, click Create.

   The Create Replication Topology tab is displayed.
4. In the Identify Servers page, enter the following information for at least two *source servers* to be configured:
   - **Host.** Enter the host name using a fully qualified domain name.
   - **Administration Port.** Enter the administration port number for the server named above. The default is 4444.
   - **Admin Username.** Enter the DN for the administrator who can manage the server.
   - **Admin Password.** Enter the password for the administrator you specified.
5. (Optional) On this page, you can also perform the following:
   - To preview the suffixes configured for a server, click its Preview Suffixes link in the last column.
   - To add another server to be configured, click **Add**, then repeat step 4 above.
   - To remove a server from the topology, select the server name, then click Remove.
6. Click **Next**.
7. In the Global Administrators page, the Domain Administrator can manage multiple directory server instances using OUDSM. This administrator is the Global Administrator who will manage the new replication topology.

   Provide the following information for the Global Administrator:
   - **Global Admin User ID.** This is the administrator who can view and manage the topology.

- **Global Admin Password.** Enter the password for the Global Administrator specified above.
- **Confirm Global Admin Password.** Enter the password again to confirm it.

Click Next.

8. In the Replication Servers page, in the Configure Replication Servers table, provide the following information for the *replication servers* to be configured:
   - **Host.** You cannot modify the server host name here.
   - **Admin Port.** You cannot modify the server administration port here.
   - **Act as Replication Server.** If you want the server to act as a replication server, then click the checkbox until a check is displayed. If you cannot modify this setting, then the server is already configured as a replication server.
   - **Replication Port.** If you enabled a server to act as a replication server in the previous field, then enter a replication port number.

   > **Note:**
   >
   > Be sure to enter a replication port number that is not already in use. If you cannot modify this setting, then the server is already configured as a replication server.

   Click **Next**.
9. In the Replication Data page, the Configure Replicated Data table displays all the suffixes that are available in at least two among the servers. Indicate whether each suffix in the topology will be replicated. The suffixes you enable here will be replicated on all the servers in the replication topology.
   - To enable a domain to act as a replication suffix, in the Configure Replicated section, select a domain from the "Available for Replication" column, then click to right arrow to move it to the "Selected for Replication" column.
   - To enable a server to act as a replication domain, click its Replicate Suffixes checkbox until a checkmark is displayed.

     Click **Next**.
10. The summary page displays the replication server and domain information you just entered.
    - If you must modify any of the displayed information, click **Back**.
    - When you are satisfied that the Summary information is correct, click **Create**.

32.3.5.2 Managing an Existing Replication Topology

To manage an existing replication topology:

1. To invoke OUDSM, enter the following URL into your browser's address field:

   `http://`*host*`:`*port*`/oudsm`

   where *host* is the name of the host on which OUDSM is running, and *port* is the port number of its administration server. The default administration port number is 7001.
2. In the Topology Manager subtab, enter the following information:

- **Host.** Enter the host name of any server that is part of the replication topology. Use the fully qualified domain name.
- **Administration Port.** Enter the administration port number for the server specified above.
- **Global Admin User ID.** Enter the Global admin User ID. This is the administrator who can view and manage the topology. The user ID was specified when the topology was created.
- **Global Admin Password.** Enter the password for the Global Administrator specified above.

Click **Connect**.

3. In the Replication topology page, you can view and manage information about the topology, and you can add additional servers to the topology.
   - To add a server to the replication topology, click **Add Servers**.
   - To automatically refresh the topology information, click the **Automatically Refresh Topology Information** checkbox until a checkmark is displayed. To manually refresh the topology information, first be sure the automatic refresh feature is disabled, then click **Refresh**.
   - To edit the value for interval after which the topology is to be automatically refreshed, click **Update**.
   - To view tasks recently executed in the replication topology, in the Launched Tasks section click the **View Launch Task Details** link.
4. In the Replication Servers and Replicated Data section, you can do to the following:
   - Use the drop-down filter lists to filter search results based on any of the following: replicated suffix, replication host name, *host:port* information, or replication group name.
   - Change the replication port number.
   - To disable replication, select the replication server or replicated suffix you want to disable. Then, in the Actions menu, choose **Disable Replication**.
5. To assign a replication server to a different replication group, in the Replication Servers section, click the **Change Replication Group** link.
6. To configure a replicated suffix on a server, in the Replicated Data section, first select the replicated suffix you want to configure, then:
   - To change the Trust/Untrust setting, click **Trust/Untrust**. For information about trusted and untrusted servers, see Understanding Isolated Replicas.

     > **Note:**
     >
     > The Trust/Untrust button will be disabled if the server used for connecting to the topology is an untrusted server.

   - To initialize the server, click **Initialize**. For information about initialization, see Understanding Replication Initialization.
   - To start pre-external initialization, click **Pre-External Initialization**. For information abut pre-external initialization, see the `pre-external-initialization` option in dsreplication, and in gicadm.

- To start post-external initialization, click **Post-External Initialization**. For information about post-external initialization, see the `post-external-initialization` option in dsreplication, and in gicadm.
- To purge historical data, click **Purge Historical**. For information about purging historical data, see Understanding How to Purge Historical Information,and Understanding Purging Historical Replication Data.
- To change the data replication group, click **Change Replication Group**. For information about replication groups, see About Replication Groups.

7. To modify the Global Administrator credentials, click **Topology Settings**. Provide the following information:
   - **Global Admin ID.** Enter the username for the administrator who can connect to and manage a topology. This username was created when the topology was created.
   - **Global Admin Password.** Enter the password for the administrator named above.
   - **Confirm Global Admin Password.** Enter again the password for the administrator name above.

   Click **Apply**.

32.4 Understanding Configuration for Large Replication Topologies

Review these topics for a contextual description of dedicated replication servers and dedicated directory servers in large topologies and how to configure a dedicated replication server.

- About Large Replicated Topologies Configuration
- Configuring a Dedicated Replication Server

32.4.1 About Large Replicated Topologies Configuration

In particularly large topologies, it is often simpler to configure *dedicated replication servers* and *dedicated directory servers*.

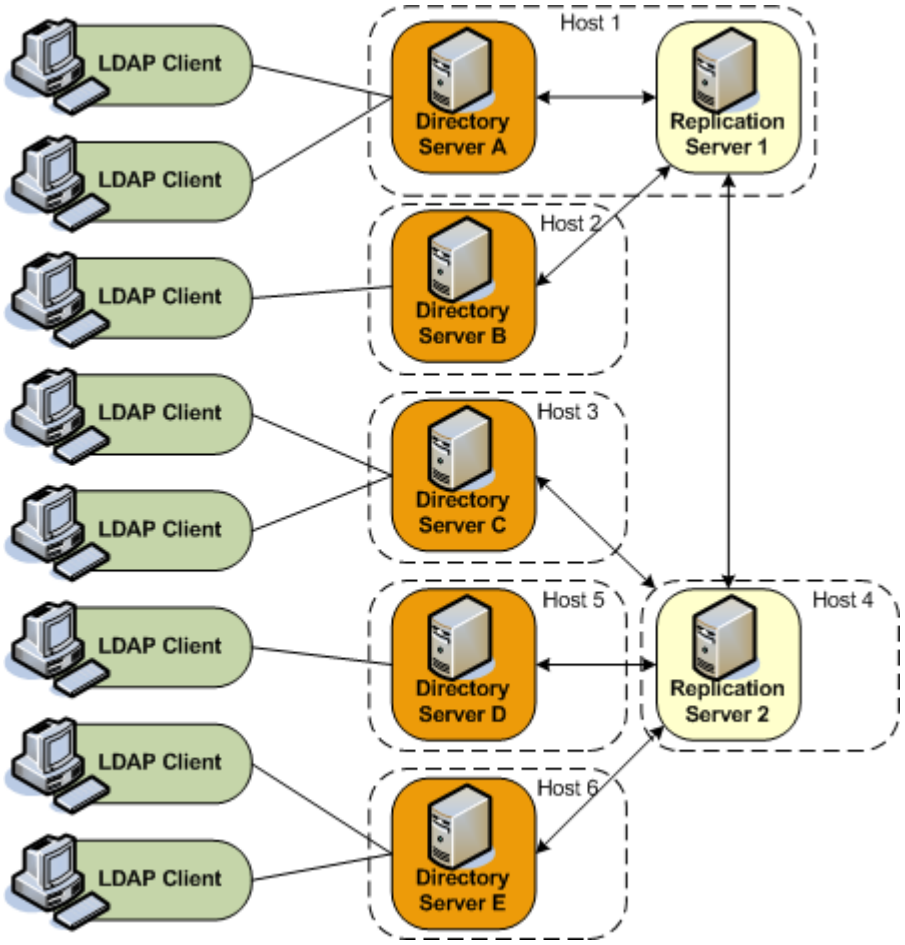
- Dedicated directory servers that do not include a directory server. They contain replicated data, but do not contain a change log with the modifications made to the replicated data. Dedicated directory servers also have no configured replication port.
- Dedicated replication servers do not include a replication server or replicated data, but they do contain a change log with the modifications made to the replicated data on other servers in the topology. Dedicated replication servers also do have a configured replication port.

> **Note:**
>
> Each topology must have at least two replication servers to avoid a single point of failure.

For more information and sample topologies, see Understanding Deployment Scenarios Using the Directory Server.

The following diagram illustrates a large replication topology with one dedicated replication server (Replication Server 2), four dedicated directory servers, and one server that contains both a replication server and a directory server (Host 1).

Figure 32-2 Large Replicated Topology



32.4.2 Configuring a Dedicated Replication Server

Use the `--onlyReplicationServer1` or `--onlyReplicationServer2` option when you enable replication between two servers to configure a dedicated replication server.

The following example configures replication between Directory Server C and Replication Server 2 in the previous illustration.

```
$ dsreplication enable \
  --host1 host3 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPasswordFile1 pwd.txt --noReplicationServer1 \
  --host2 host4 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 pwd.txt --onlyReplicationServer2 \
  --replicationPort2 8989 --adminUID admin --adminPasswordFile pwd.txt \
  --baseDN "dc=example,dc=com" -X -n
```

32.5 Modifying the Replication Configuration With `dsconfig`

You can change certain advanced properties of a replication configuration by using the `dsconfig` command. Advanced properties are usually optional, or have a default value that is acceptable in most cases.

For general information about using `dsconfig`, see Managing the Server Configuration Using `dsconfig`.

You cannot use `dsconfig` to set up replication between directory servers. Replication can be set up automatically using the GUI install utility, or manually, using the `dsreplication` command. See Configuring Data Replication Using OUDSM.

This section covers the following topics:

- Retrieving the Replication Domain Name
- Configuring Replication Purge Delay
- Configuring Window Size
- Configuring Initialization Window Size
- Configuring Heartbeat Interval
- Changing the Isolation Policy
- Configuring Encrypted Replication
- Configuring Replication Groups
- Configuring Assured Replication
- Configuring Fractional Replication
- Configuring Replication Status
- Configuring the Replication Server Weight

32.5.1 Retrieving the Replication Domain Name

The *replication domain name* is generated by the directory server and includes the base DN and a numeric unique identifier.

To obtain a list of the configured replication domains, use the `list-replication-domains` subcommand. For example:

```
$ dsconfig -h host1 -p 4444 -D "cn=directory manager" -j pwd-file -n list-replication-domains \
   --provider-name "Multimaster Synchronization"

Replication Domain : Type    : server-id : replication-server     : base-dn
-------------------:---------:-----------:------------------------:--------------------
cn=admin data      : generic : 13981     : host1:8989, host2:8989 : cn=admin data
cn=schema          : generic : 20284     : host1:8989, host2:8989 : cn=schema
dc=example,dc=com  : generic : 26560     : host1:8989, host2:8989 : "dc=example,dc=com"
```

32.5.2 Configuring Replication Purge Delay

The replication changes database maintains a record of updates, which might or might not have been replicated. The replication purge delay is a property of the replication server, and

specifies the period of time after which internal purge operations are performed on the replication changes database.

This section covers the following topics:

- How Replication Changes Are Purged
- Changing the Replication Purge Delay

32.5.2.1 How Replication Changes Are Purged

Any change that is older than the purge delay is removed from the replication changes database, irrespective of whether that change has been applied. The default purge delay is 100 hours. If the replication changes database is backed up less frequently than the purge delay, changes will be cleared before the changes database has been backed up. Changes can therefore be lost if you use the backup to restore data.

32.5.2.2 Changing the Replication Purge Delay

To change the replication purge delay:

1. Display the current value of the replication purge delay.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property replication-purge-delay

Property               : Value(s)
-----------------------:---------
replication-purge-delay : 1 d
```

2. Change the purge delay.

   The following command changes the purge delay to one week:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" \
  --set replication-purge-delay:1w
```

32.5.3 Configuring Window Size

The window size is a property of the replication server and specifies the number of change requests that are sent to directory servers, without the replication server having to wait for an acknowledgment from the directory server before continuing.

This section covers the following topics:

- About Window Size.
- Changing the Window Size.

32.5.3.1 About Window Size

The window size represents the maximum number of update messages that can be sent without immediate acknowledgment from the directory server. It is more efficient to send many messages in quick succession instead of waiting for an acknowledgment after each one. Using the appropriate window size, you can eliminate the time replication servers spend waiting for acknowledgments to arrive. The default window size is 100. If you notice that some directory

servers are lagging behind in terms of replicated changes, increase the window size to a higher value and check replication performance again before making further adjustments.

32.5.3.2 Changing the Window Size

To change the window size:

1. Display the current value of the window size:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-replication-server-prop --provider-name "Multimaster Synchronization" \
   --advanced --property window-size

Property    : Value(s)
------------:---------
window-size : 100
```

2. Change the window size.

   The following command changes the window size to 200.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --set window-size:200
```

32.5.4 Configuring Initialization Window Size

During a data import in a replicated topology, it can occur that the importing server is too slow to keep up with the data that is sent by the exporting server. The importing server can therefore block not only the import, but can also stop any other replication changes from being propagated by the exporting server.

This section covers the following topics:

- About Initialization Window Size.
- Changing the Initialization Window Size.

32.5.4.1 About Initialization Window Size

An initialization window size enables an exporting server to detect acknowledgments from the slowest importing server and to send data on the replication network *only* when the slow importer is available to receive them.

The initialization window size is set to `100` by default. If there are no slow servers in your topology, you can increase the initialization window size so that exporting servers send more updates before waiting for an acknowledgment. If your topology includes a particularly slow server, you can decrease the initialization window size to ensure that replication is not blocked by this server.

32.5.4.2 Changing the Initialization Window Size

To change the initialization window size:

1. Display the current value of the initialization window size:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name dc=example,dc=com --advanced --property initialization-window-size
Property                  : Value(s)
```

```
--------------------------:---------
initialization-window-size : 100
```

2. Change the initialization window size.

   The following command changes the initialization window size to 50.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name dc=example,dc=com --set initialization-window-size:50
```

32.5.5 Configuring Heartbeat Interval

The heartbeat interval is a property of the replication domain and specifies the frequency with which the replication domain communicates with the replication server. The replication domain expects a regular heartbeat at this interval from the replication server. If the heartbeat is not received, the domain closes its connection and connects to another replication server in the topology.

This section covers the following topics:

- About Heartbeat Interval.
- Changing the Heartbeat Interval.

32.5.5.1 About Heartbeat Interval

The default heartbeat interval is ten seconds. If replication is running over a WAN or a network with slow response times, you might want to increase the heartbeat interval. In addition, if you observe an error similar to the following in the logs, it is probably necessary to increase the heartbeat interval.

```
[26/May/2011:16:32:50 +0200] category=SYNC severity=NOTICE msgID=15138913
 msg=Replication Heartbeat Monitor on RS rserver/192.157.197.62:8989 30382 for
 dc=example,dc=com in DS 10879 is closing the session because it could not
 detect a heartbeat
```

The heartbeat interval is sensitive to the settings of your JVM. If you require a lower heartbeat interval than the default, you must configure your JVM to have a low pause time during garbage collection by setting the `-XX:+UseConcMarkSweepGC` option. For more information, see "Configuring the JVM, Java Options, and Database Cache" in *Installing Oracle Unified Directory*.

32.5.5.2 Changing the Heartbeat Interval

To change the heartbeat interval:

1. Display the current value of the heartbeat interval.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" --advanced \
  --property heartbeat-interval
Property           : Value(s)
-------------------:---------
heartbeat-interval : 10 s
```

2. Change the heartbeat interval.

   The following command changes the heartbeat interval to 5 seconds.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" --set heartbeat-interval:5s
```

32.5.6 Changing the Isolation Policy

The isolation policy is a property of the replication domain and specifies the behavior of the directory server if replication is configured but none of the replication servers are up and running when an update is received. The default behavior of the directory server in this situation is to reject all updates.

To change the isolation policy:

1. Display the current isolation policy.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
 get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" \
  --advanced --property isolation-policy -n

Property         : Value(s)
-----------------:------------------
isolation-policy : reject-all-updates
```

2. Change the isolation policy.

   The following command specifies that the directory server should accept all updates in this situation.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" \
  --set isolation-policy:accept-all-updates -n
```

32.5.7 Configuring Encrypted Replication

By default, replication traffic is not encrypted. To enable encryption, use the `dsconfig` command to set the properties of the crypto manager.

The following command specifies that replication traffic should be encrypted.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-crypto-manager-prop --set ssl-encryption:true
```

32.5.8 Configuring Replication Groups

Review these topics for a contextual description of Replication Group and the instructions to configure a replication group.

- About Replication Group
- Configuring a Replication Group

32.5.8.1 About Replication Group

Replication groups are designed to support multi-data center deployments and disaster recovery scenarios.

A replication group is configured on each directory server and replication server that should be part of the same group. On directory servers, a replication group is configured *per replicated domain*. On replication servers, the group is configured for the entire replication server. For information about the design and implementation of replication groups in the directory server, see About Replication Groups.

> **Note:**
>
> Changing the replication group configuration has an impact on assured replication. For more information, see Understanding Assured Replication.

32.5.8.2 Configuring a Replication Group

Replication groups are configured by giving each replicated domain and replication server the same group ID. This example configures a replication group (1) for the replicated domain `dc=example,dc=com`.

1. On each directory server that will be part of this group, set the group ID for the domain `dc=example,dc=com`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set group-id:1
```

2. On each replication server that will be part of this group, set the group ID.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set group-id:1
```

32.5.9 Configuring Assured Replication

Assured replication is a method of making regular replication work in a more synchronized manner.

This section covers the following topics:

- About Assured Replication Configuration
- Configuring Assured Replication in Safe Data Mode
- Configuring Assured Replication in Safe Read Mode

32.5.9.1 About Assured Replication Configuration

In most deployment scenarios, the loosely consistent multi-master replication model is sufficient. However, certain scenarios might require tighter consistency between replicas. In such cases, you can configure *assured replication*, which provides the following benefits:

- **High availability of data.** If a server crashes immediately after a modification is received on that server, there is a risk that the modification will be lost before it is replayed to other servers in the topology. With assured replication, any modification is replayed to another server in the topology *before* an acknowledgment is sent to the client application., which minimizes the risk of losing data if the server crashes.

- **Immediacy of data availability.** Some applications might require modifications to be available on additional servers in the topology immediately after a modification is made.

Assured replication is an extension of the replication protocol and is configured *per replicated domain*. For more information, see Retrieving the Replication Domain Name.

Assured replication is different from *synchronous replication*. That is, changes do not occur simultaneously on all servers in the topology. However, assured replication can mimic the functionality of synchronous replication to an extent, as far as LDAP clients are concerned. This is achieved by delaying acknowledgments to the client application until a modification has been propagated to additional servers in the topology.

> **Note:**
>
> Assured replication relies on *replication groups*. All replication servers and directory servers that function together in an assured replication configuration must be part of the same replication group.

Assured replication can function in two modes:

- **Safe data mode.** Any update must be propagated to a defined number of replication servers before the client receives an acknowledgment that the update has been successful.

  The number of replication servers that must be reached defines the *safe data level*. The higher the safe data level, the higher the overall data availability.
- **Safe read mode.** Any update must be propagated to all the directory servers in the topology before the client receives an acknowledgment that the update has been successful.

In both safe data mode and safe read mode, you can configure a time-out interval to prevent LDAP client calls from hanging if certain servers in the topology are not available.

- On each *directory server*, you can configure a global time-out that comes into effect when the directory server sends an update to its replication server, either safe data mode or safe read mode. If this time-out is reached, the LDAP client call returns immediately and a message is written to the replication log to track the event.
- On each *replication server*, you can configure a global time-out that comes into effect when the replication server sends an update to a peer replication server or to another directory server, either in safe data mode or in safe read mode. If this time-out is reached, the acknowledgment message that is returned to the initiating server (either a directory server or a replication server) includes a message that indicates the time-out. The initial directory server then logs a message that the time-out occurred for that update.

> **Note:**
>
> The default time-out of two seconds for a directory server and one second for a replication server should be satisfactory for most deployments. *Only* change the time-out if you are viewing time-outs in the logs and if you have a complete understanding of the impact of such a change. The value of the time-out should reflect the anticipated time that an update requires to go through its full path to reach its destination.
>
> The time-out value on a directory server should always be higher than the value on the replication server. For example: `DS1(timeout 2s) -> RS1(timeout 1s) -> RS2(timeout 1s) -> DS2`.

For a detailed explanation of the assured replication mechanism and the various configurable options, see Understanding Assured Replication.

32.5.9.2 Configuring Assured Replication in Safe Data Mode

This procedure configures assured replication in safe data mode for a topology. The procedure assumes that replication has already been configured.

1. On each directory server in the topology:
   a. Set the assured replication mode.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-type:safe-data
```

   b. Set the safe data level.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-sd-level:2
```

   If you have configured replication by using `setup` or `dsreplication`, your replication servers and directory servers will be on the same virtual machine. In this case, you must set the safe data level to 2 or higher.

   c. Set the assured replication time-out.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced\
  --set assured-timeout:5s
```

   Only change the time-out if you are viewing time-outs in the logs and if you have a complete understanding of the impact of such a change.

   d. Verify the directory server group ID.

   This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see Configuring Replication Groups.

e. Display the current assured replication configuration.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --property assured-type --property assured-sd-level --property assured-timeout

Property         : Value(s)
-----------------:------------
assured-sd-level : 2
assured-timeout  : 5 s
assured-type     : safe-data
```

2. On each replication server in the topology:

a. Display the current assured replication configuration.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property assured-timeout --property group-id

Property                  : Value(s)
--------------------------:---------
assured-timeout           : 1 s
group-id                  : 1
```

b. Set the assured replication time-out.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set assured-timeout:5s
```

Only change the time-out if you are viewing time-outs in the logs and if you have a complete understanding of the impact of such a change.

c. Verify the replication server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see Configuring Replication Groups

32.5.9.3 Configuring Assured Replication in Safe Read Mode

Assured replication is configured *per replicated domain*. This procedure configures assured replication in safe read mode for a topology. The procedure assumes that replication has already been configured.

1. On each directory server in the topology:

a. Set the assured replication mode.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-type:safe-read
```

b. Set the assured replication time-out.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
```

```
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-timeout:5s
```

Only change the time-out if you are viewing time-outs in the logs and if you have a complete understanding of the impact of such a change.

c. Verify the directory server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see Configuring Replication Groups. For more information about groups and assured replication, see Understanding Assured Replication.

d. Display the current assured replication configuration.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --property assured-type --property assured-timeout --property group-id

Property         : Value(s)
-----------------:------------
assured-timeout  : 5 s
assured-type     : safe-read
group-id         : 1
```

2. On each replication server in the topology:

a. Display the current assured replication configuration.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property assured-timeout --property degraded-status-threshold \
  --property group-id

Property                  : Value(s)
--------------------------:---------
assured-timeout           : 1 s
degraded-status-threshold : 5000
group-id                  : 1
```

b. Set the assured replication time-out.

Only change the time-out if you are viewing time-outs in the logs and if you have a complete understanding of the impact of such a change.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set assured-timeout:5s
```

c. Set the degraded status threshold.

The degraded status threshold defines the stage at which the server is regarded as "too slow", based on the number of updates queued in the replication server for that directory server. For more information, see What is Degraded Status?.

Do *not* adjust this value unless you observe time-outs in the logs.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set degraded-status-threshold:2000
```

d. Verify the replication server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see Configuring Replication Groups. For more information about groups and assured replication, see Understanding Assured Replication.

32.5.10 Configuring Fractional Replication

The following topics describe how to configure fractional replication on one or more servers in a topology:

- About Fractional Replication Configuration
- Configuring Exclusive Fractional Replication
- Configuring Inclusive Fractional Replication
- Configuring and Initialize a Fractional Domain

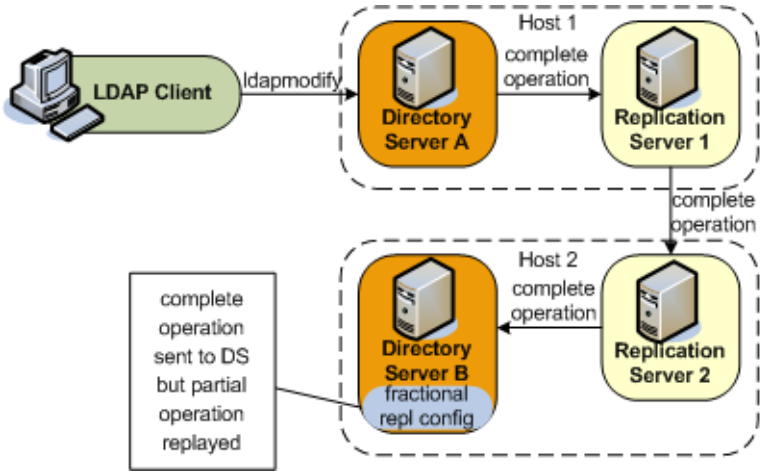
For information about the architecture of the fractional replication mechanism, see Overview of Fractional Replication.

32.5.10.1 About Fractional Replication Configuration

Fractional replication enables you to replicate specific parts of directory data to other replicas in the topology. This feature is particularly useful in the following scenarios:

- **Limited disk space.** Restricting the data that is replicated can significantly cut down on the amount of disk space that is required on certain replicas, particularly if you restrict the replication of attributes such as jpeg photos, which represent large data volumes.
- **Security concerns.** Certain data, such as user passwords, might be sensitive and not required on certain replicas, especially if there is a risk of inappropriate access on these replicas.

Fractional replication is configured on the directory server that receives the partial data, and is attribute-based. Consider the following illustration:



Fractional replication is configured on Directory Server B. An `ldapmodify` operation is sent to Directory Server A. The entire operation is forwarded to Replication Server 1, then to Replication Server 2, then to Directory Server B. When the operation is replayed on Directory

Server B, certain attributes from the operation are filtered out, based on that server's fractional configuration.

Fractional replicas remain writable directly from client applications. However, if an add or modify operation that includes certain "forbidden attributes" is attempted on a fractional replica, the operation is denied and the server returns an "Unwilling to perform" error.

You can configure fractional replication in one of two modes:

- **Exclusive mode.** In this mode, the multi-valued `fractional-exclude` attribute is used to filter out the specified attributes from an incoming LDAP add or modify operation.

  Excluded attributes must be *optional* attributes of an object class.

- **Inclusive mode.** In this mode, the multi-valued `fractional-include` attribute is used to filter in only the specified attributes from an incoming LDAP add or modify operation.

  All other attributes (except for those that are mandatory in the object class) are removed from the change that is replayed on the server.

The two modes are mutually exclusive, that is, you can include only one of these attributes in a domain configuration.

Fractional replication is configured *per replicated domain* (see Retrieving the Replication Domain Name). A *fractional domain* implies that certain attributes are entirely absent from the domain. These attributes are filtered out at operation replay time but are also absent from the existing data in the domain.

To ensure coherency of the data across a replicated topology, it is necessary to identify whether a particular data set is fractional. The configuration of a new fractional domain therefore implies specific steps to ensure that the domain is free of forbidden attributes, and recognizable as a fractional domain. For more information, see Configuring and Initialize a Fractional Domain.

Use the `dsconfig` command to configure fractional replication in a domain, as follows.

32.5.10.2 Configuring Exclusive Fractional Replication

The following example configures a replica to exclude the `photo` and `jpegPhoto` attributes from any creation or modification of an entry whose object class is `inetOrgPerson`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-exclude:inetOrgPerson:photo,jpegPhoto
```

Object classes and attributes can be specified by their names, or by their OIDs, so the following example has the same effect as the previous example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-exclude:2.16.840.1.113730.3.2.2:0.9.2342.19200300.100.1.7, \
  0.9.2342.19200300.100.1.60
```

If you use object class or attribute names *and* OIDs, both values are added. For example, the following command adds both the attribute name and its OID to the list of excluded attributes:

```
$ dsconfig set-replication-domain-prop ...
  --set fractional-exclude:*:jpegPhoto,*:0.9.2342.19200300.100.1.60
```

If you wanted to remove this attribute from the list, you would need to remove both the attribute name and the OID.

To specify that the `photo` and `jpegPhoto` attributes should be removed from any creation or modification of any entry (regardless of its object class), use an asterisk in place of the object class. For example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-exclude:*:photo,jpegPhoto
```

32.5.10.3 Configuring Inclusive Fractional Replication

The following example configures a replica to include only the `uid` and `employeeNumber` attributes from any creation or modification of an entry whose object class is `inetOrgPerson`. All other attributes are ignored in the modification, except those that are mandatory for the object class.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-include:inetOrgPerson:uid,employeeNumber
```

Object classes and attributes can be specified by their names, or by their OIDs, so the following example has the same effect as the previous example:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-include:2.16.840.1.113730.3.2.2:0.9.2342.19200300.100.1.1, \
  2.16.840.1.113730.3.1.3
```

If you use object class or attribute names *and* OIDs, both values are added. For example, the following command adds both the attribute name and its OID to the list of included attributes:

```
$ dsconfig set-replication-domain-prop ...
  --set fractional-include:*:jpegPhoto,*:0.9.2342.19200300.100.1.60
```

If you wanted to remove this attribute from the list, you would need to remove both the attribute name and the OID.

To specify that a particular attribute should be included in the creation or modification of any entry (regardless of its object class), use an asterisk in place of the object class. The following example includes only the description attribute in a creation or modification operation on any entry.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" \
  --set fractional-include:*:description
```

32.5.10.4 Configuring and Initialize a Fractional Domain

To initialize a new fractional domain:

1. Configure exclusive or inclusive fractional replication, as described in the previous two sections.

   At this point, the domain obtains a *bad generation ID* status. For more information, see Overview of Replication Status.

This means that all modifications on the domain are blocked until the data is synchronized with the rest of the topology.

2. Import a new data set from one of the other servers in the topology.

   The new data set can be imported online, by using `dsreplication initialize` or by using `import-ldif` in online or offline mode. The server from which you import the data must either be an entire replica (that is, not a fractional replica) or must have the same fractional configuration as the server to which you are importing the data. During the import, all entries will be filtered with the fractional configuration set up in the previous step.

   For information about how to import a data set, see Initializing a Single Replicated Server and Importing and Exporting Data.

3. After the data import, the domain returns to *normal* status.

   For more information, see Overview of Replication Status.

   The domain is now able to accept new entries from local LDAP operations, or synchronization operations with other servers in the topology. The data in the domain is free of any "forbidden" attributes.

32.5.11 Configuring Replication Status

Each replicated domain in a replicated topology has a certain *replication status*, depending on its connections within the topology, and on how up to date it is regarding the changes that have occurred throughout the topology.

For more information, see Overview of Replication Status.

This section covers the following topics:

- About Configuration of Degraded Status Threshold Parameter in Replication Status.
- Configuring the Degraded Status Threshold.

32.5.11.1 About Configuration of Degraded Status Threshold Parameter in Replication Status

Replication status is generated automatically, based on how up-to-date a server is within the replicated topology. The only configurable parameter is the degraded status threshold, which defines the maximum number of changes allowed in the replication server's queue for all domains of the directory servers that are connected to this replication server. When this number is reached for a specific directory server, that server is assigned a degraded status. The degraded status remains until the number of changes drops below this value.

> **Note:**
>
> The default value of the degraded status threshold should be adequate for most deployments. Only modify this value if you observe several time-out messages in the logs when assured replication is configured.

32.5.11.2 Configuring the Degraded Status Threshold

The default number of changes defined by this threshold is 5000. This example sets the threshold to 6000, to account for a network with more latency.

On the replication server, use `dsconfig` to set the degraded status threshold.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-replication-server-prop --provider-name "Multimaster Synchronization" \
  --set degraded-status-threshold:6000
```

32.5.12 Configuring the Replication Server Weight

In large topologies with several directory servers and several replication servers, it is more efficient to spread the directory servers out across the replication servers in a predefined manner.

You can specify how many directory servers should connect to each replication server in a topology according to the relative capacity of the machine on which the replication server is running. For more information, see Understanding Replication Server Load Balancing.

To configure the replication server weight, run the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --set weight:2
```

By default, the weight of each replication server in the topology is 1.

32.6 Initializing a Replicated Server With Data

You can initialize replicated server by using the `gicadm` command, which accesses the server configuration over SSL through the administration connector.

The following topics describe how to initialize a replicated server with data:

- Initializing a Single Replicated Server
- Initializing a New Replicated Topology
- Adding a Directory Server to an Existing Replicated Topology
- Changing the Data Set in an Existing Replicated Topology
- Appending Data in an Existing Replicated Topology

> **Note:**
>
> For more information, see Managing Administration Traffic to the Server and gicadm.
>
> In addition, because this section references information provided in Populating a Stand-Alone Directory Server With Data, be sure to read that section before you proceed.

32.6.1 Initializing a Single Replicated Server

The easiest way to initialize a single directory server in a replicated topology is to use the `dsreplication` command to copy the data over from another directory server in the topology. This command requires replication to have been enabled between the source server and the

destination server. The command replaces all data under the specified base DN on the destination server with the data from the source server.

For example, the following command initializes the base DN "dc=example,dc=com" on host2 with the data on host1.

```
$ dsreplication initialize --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPasswordFile pwd.txt \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 --trustAll
```

32.6.2 Initializing a New Replicated Topology

In a new replicated topology, you can either initialize a single directory server or all directory servers individually with the same data.

- Initialize all directory servers individually with the same data, using one of the methods described in Populating a Stand-Alone Directory Server With Data. When you have initialized all directory servers with data, enable replication between the servers.
- Initialize a single directory server using one of the methods described in Populating a Stand-Alone Directory Server With Data. Enable replication for all directory servers, then use the dsreplication intialize-all command to initialize all the remaining servers simultaneously. This command takes the details of the source server as arguments, and initializes all other servers for which replication is enabled.

  For example, the following command initializes all directory servers from the contents on host1.

  ```
  $ dsreplication initialize-all --hostname localhost --port 4444 --trustAll \
    --baseDN "dc=example,dc=com" --adminUID admin --adminPasswordFile pwd.txt
  ```

32.6.3 Adding a Directory Server to an Existing Replicated Topology

When you add a directory server to an existing replicated topology, the new server must be populated with the same *generation* of data as the existing directory servers in the topology. The data generation is an ID stored within the root entry of the replication domain. When the data generation does not exist, it is computed by the replication mechanism and stored.

To ensure that the new directory server has the same data generation as the other servers in the topology, use one of the following methods to populate the directory server with data:

- Use the same original LDIF file, backup file, or binary copy that was used to populate the other directory servers.
- Use the result of an export, backup, or binary copy from another directory server in the topology.

If you install the new directory server using the GUI install and specify that it will be part of the replicated topology, the server is initialized with the correct data generation automatically.

If you do not install the directory server using the GUI install, and you use the dsreplication command to enable replication, you must initialize the server manually using one of the methods described in the previous section.

If a directory server in the topology does not contain the same data generation as the rest of the topology, data cannot be replicated to or from the server. However, the directory server remains connected to the topology, enabling it to be initialized using the replication protocol. Replication on this directory server is said to be *downgraded*.

When a directory server with the correct data generation is added to an existing topology, the replication mechanism automatically replays any changes that occurred since the first directory server in the topology was initialized with data. This action ensures that the new directory server is synchronized with the rest of the topology.

32.6.4 Changing the Data Set in an Existing Replicated Topology

Changing the data set implies importing an entirely new set of data to every directory server in the topology.

When the data set is changed, two tasks are performed:

- The new data is applied to each directory server in the topology.
- The replication servers are cleared of any changes they might contain. This task includes resetting the data generation on the directory servers so that the new data generation is used.

If you change the data set using the `dsreplication initialize` command, both of these tasks are performed automatically. However, if you use the `import-ldif` command or the binary copy method to change the data set, you must perform these tasks manually.

To change the data set with `import-ldif`, Binary copy or `restore`:

1. Clear the generation ID from the directory servers by running the `dsreplication pre-external-initialization` command.

   It is sufficient to run this command from any directory server in the topology. All directory servers in the topology will be updated. For example, the following command prepares all servers in the topology for initialization by using `import-ldif` or binary copy:

   ```
   $ dsreplication pre-external-initialization -h host1 -p 4444 -X \
     -b dc=example,dc=com -I admin -j pwd-file

   Establishing connections and reading configuration ..... Done.

   pre-external-initialization should only be used if you are going to initialize
   all the replicated servers. If it is not the case (for instance you are going
   to recover only a server or you are in the process of adding a new server to
   the replication topology), the subcommand must not be executed.
   Do you want to continue? (yes / no) [yes]:
   ```

2. Use import-ldif, binary copy or restore to initialize all directory servers in the topology with data. When initializing the data on a directory server with either `import-ldif`, `binary copy` or `backup-restore`, you can observe following behaviors for changelog:

   - When you use `pre-external-initialization` command, the changelog is reset , i.e. any values existing in the `changelogDB` prior to running the `pre-external-initialization` command are cleared. Thus, using the `import-ldif` command or the `binary copy`(only if we copy <instance>/OUD/db) mechanism will not restore any `changelogDB` existing prior running the `pre-external-initialization` command.
   - The `changelogDB` is restored back only when full backup is done using the backup command before running the `pre-external-initialization` command. The complete backup is restored using the `restore` command along with the replicationChanges which exists in backed up directory.

> **Note:**
>
> If the `restore` command is run and the replicationChanges are not restored, then the backedup changelogs will be lost.

3. Reset the generation ID by running the `dsreplication post-external-initialization` command.

   It is sufficient to run this command from any directory server in the topology. All other directory servers are updated. For example, the following command resets the generation ID for all directory servers in the topology after initialization using `import-ldif` or binary copy:

```
$ dsreplication post-external-initialization -h localhost \
  -p 4444 -b dc=example,dc=com -I admin -j pwd-file -X
Updating replication information on base DN dc=example,dc=com ..... Done.
Post initialization procedure completed successfully.
```

32.6.5 Appending Data in an Existing Replicated Topology

The easiest way to import a large number of entries to an existing replicated topology that already contains a large number of entries is to use the `import-ldif` command with the `-a` or `--append` option.

When you import data by using the `import-ldif` command, the imported data is not replicated automatically. You must therefore run `import-ldif --append` on every directory server in the topology. This strategy enables you to import the data with no downtime in the directory service.

You can also use the `dsreplication initialize-all` command after you have imported the data to a single directory server in the topology. However, this strategy will result in the directory service being unavailable for a certain period of time.

32.7 Using the External Change Log

The External Change Log (ECL) publicizes all changes that have occurred in a directory server database and is particularly useful for synchronizing the LDAP directory with other subsystems.

The ECL is built online from the replication change log and does not use an additional database for its storage. It is not a regular JEB back end, therefore no index must be configured.

The following topics describe how to enable the ECL in your directory service and how to configure client applications so that they can access the ECL:

- Enabling the External Change Log
- About External Change Log APIs
- How a Client Application Uses the External Change Log in Cookie Mode
- Format of External Change Log Entries
- Specifying the Attributes to be Included in the External Change Log
- Specifying the Attributes to be Excluded in the External Change Log
- Initializing Client Applications to Use the External Change Log

32.7.1 Enabling the External Change Log

The External Change Log (ECL) is available by default on any server instance that includes *both* a directory server *and* a replication server. The ECL is enabled when a directory server is configured as part of a replicated topology during installation or when replication is configured after installation.

The ECL is not available by default on a server instance that is configured as either a *dedicated directory server* or a *dedicated replication server* (as described in Understanding Configuration for Large Replication Topologies).

The ECL is enabled when replication is configured in one of the following ways:

- By configuring a directory server as part of a replicated topology during installation. For more information, see "Setting Up Replication During Installation" in *Installing Oracle Unified Directory*.
- By configuring replication after installation, by using the `dsreplication` command. For more information, see Configuring Data Replication Using OUDSM.

> **Note:**
>
> The ECL is *not* available if you configured replication with the `--onlyReplicationServer` or `--noReplicationServer` options.

Although the ECL functionality is based on the replication mechanism, some client applications might require access to the ECL content on a local server, outside of a replicated topology. You can enable the ECL on a local server, for a specific base DN, by running the following command:

```
$ dsreplication enable-changelog -h localhost -p 4444 -D "cn=directory manager" \
  -j pwd-file -r 8989 -b dc=example,dc=com -X -n
```

The replication port (`-r`) is required to configure the ECL, even on a standalone server, because the ECL relies on the replication mechanism. You need only specify the replication port if the change log (or replication) was not previously configured on the server. The default value of the replication port is 8989.

To verify that the ECL is configured on a directory server instance, run the following search command:

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
  -s base -b "" "objectclass=*" namingContexts
dn:
namingContexts: cn=changelog
namingContexts: dc=Europe,dc=com
namingContexts: dc=us,dc=com
```

32.7.2 About External Change Log APIs

The External Change Log supports two APIs, which enable two distinct *modes* of operation.

The two distinct *modes* of operation are:

- **Cookie mode.** This is the recommended API that you should use to access the ECL.

  In cookie mode, the client application provides an ECL exchange control in its request to the server. In this mode, the DIT and schema provided in the entries that are returned by the server are not compatible with the LDAP change log draft (`http://tools.ietf.org/html/draft-good-ldap-changelog-04`).

- **Draft-compatible mode.** This mode should be used only by existing applications that rely on the LDAP change log draft.

  In this mode, the DIT and schema provided in the entries that are returned by the server are compatible with the LDAP change log draft.

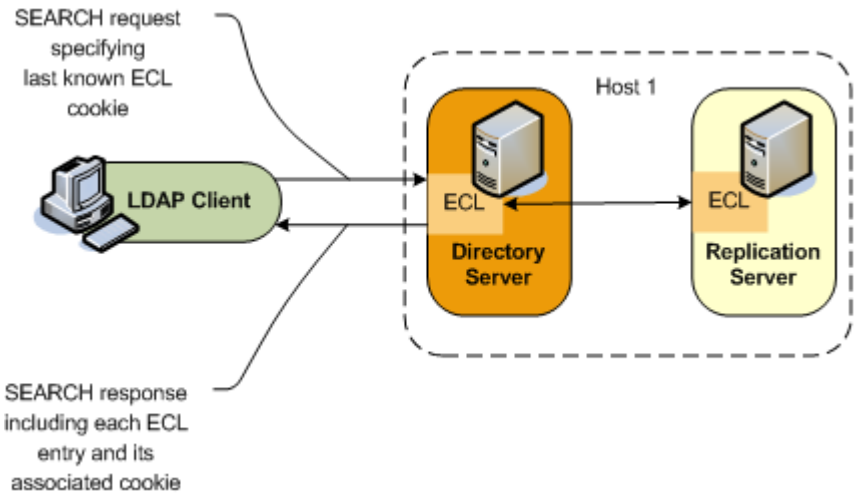
  For improved performance and for simplicity, you should port client applications to use the cookie mode. For more information, see Porting Applications that Rely on Other Change Logs.

32.7.3 How a Client Application Uses the External Change Log in Cookie Mode

Each entry in the ECL has an associated cookie. When a client application sends a SEARCH request, the application provides either the cookie of the last message that was read from the ECL (in a previous SEARCH), or an empty value. The server returns the ECL entries associated with that cookie.

Each entry is returned with its associated cookie. When the application disconnects, it stores the last cookie that it received, and provides this cookie to the server with its next SEARCH request.

This transmission of ECL cookies is illustrated in the following diagram.



The content of the cookie is *not* a public interface for the client application. The client application sends the cookie as a request control and the server sends the cookie as a response control.

The External Changelog Cookie Control has an OID of `1.3.6.1.4.1.26027.1.5.4`. If the server identifies that the cookie provided by the application is corrupted, the request is rejected. The request is also rejected if the server identifies that the configuration of the ECL has changed since the server sent this cookie to the application, or that the ECL has been purged and the oldest change stored is newer than the cookie value. In this case, additional information is returned, indicating that a full re synchronization of the external application is recommended.

> **Note:**
>
> If a server is disconnected from the replication topology and processes changes from clients that are connected to it, convergence cannot be guaranteed.

The following request and response examples indicate how the client application searches using the external change log and how the ECL responds.

Request One
To start reading the ECL, the client sends the first SEARCH request on `cn=changelog`, specifying an empty value in the External Changelog Cookie Control.

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
  --control "1.3.6.1.4.1.26027.1.5.4:false:;" -b "cn=changelog" \
  "(objectclass=*)" "*" +
```

Response One
The server sends each change to the client in a `SearchResultEntry`. The cookie attribute specifies the new cookie value. This value is also sent in a External Changelog Cookie Control, along with the entry.

```
# Public changelog exchange control(1.3.6.1.4.1.26027.1.5.4):
  dc=europe,dc=com:0000012187eae081456200000001;o=example:;
dn: replicationcsn=0000012187eae081456200000001,dc=europe,dc=com,cn=changelog
objectClass: top
objectClass: changeLogEntry
replicationCSN: 0000012187eae081456200000001
replicaIdentifier: 17762
targetDN: cn=chek-piao chea,ou=unit1,o=people,dc=europe,dc=com
changeTime: 20090528155105Z
changes:: cmVwbGFjZTogc2VlQWxzbwpzZWVBbHNvOiBjbj1tY29uZmlnCi0KcmVwbGFjZTogbW9kaW
  ZpZXJzTmFtZQptb2RpZmllcnNOYW1lOiBjbj1EaXJlY3RvcnkgTWFuYWdlcixjbj1Sb290IEROcyxjb
  j1jb25maWcKLQpyZXBsYWNlOiBtb2RpZnlUaW1lc3RhbXAKbW9kaWZ5VGltZXN0YW1wOiAyMDA5MDUy
  ODE1NTEwNVoKLQo=
changeType: modify
changeLogCookie: dc=europe,dc=com:0000012187eae081456200000001;
targetEntryUUID: 08d1830c-02f1-34a6-9cf4-8d1270ec1db0
changeNumber: 0
```

Request Two
To read the ECL from the last returned entry, the client sends the SEARCH request on `cn=changelog`, specifying the last cookie value that it received in the External Changelog Cookie Control.

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file
  --control
"1.3.6.1.4.1.26027.1.5.4:false:dc=europe,dc=com:0000012187eae081456200000001;"
  -b "cn=changelog" "(objectclass=*)"
```

> **Note:**
>
> The contents of the external change log are base 64 encoded. For information about decoding the content, see base64.

32.7.4 Format of External Change Log Entries

You can review the DN format for entries that are returned in the external change log.

`replicationcsn=`*`replicationCSN`*`,`*`replication-domain-DN`*`,cn=changelog`

For example:

```
dn: replicationcsn=0000012187eae081456200000001,dc=europe,dc=com,cn=changelog
```

The following attributes are returned for ECL entries:

```
targetDN / MUST
changeType / MUST
changeTime / MUST
changeNumber / MUST // used only for compatibility mode

changes / MAY, MUST for add, mod
newRDN / MAY, MUST for modrdn
deleteOldRDN / MAY, MUST for modrdn
newSuperior / MAY, MUST for modrdn

replicaIdentifier / MAY, OPERATIONAL / specific OUD value
replicationCSN / MAY, OPERATIONAL / specific OUD value
targetEntryuuid / MAY, OPERATIONAL / specific OUD value
changelogcookie / MAY, OPERATIONAL
```

32.7.5 Specifying the Attributes to be Included in the External Change Log

By default, attributes are included in the ECL only if they are affected by a change operation. So, for example, if the `sn` attribute of an entry is modified, only that attribute will appear in the ECL. You can, however, specify a list of attributes that will be included in the ECL regardless of whether they are affected by a change operation. In addition, you can also determine if this list of attributes is included for all types of operations or for delete operations only.

You can configure the attributes using the **`ecl-include`** property or **`ecl-include-del-only`** property. For instructions on configuring the attributes using these properties, see the following sections:

- Configuring the Attributes Using the **`ecl-include`** Property.
- Configuring the Attributes Using the **`ecl-include-del-only`** Property.

32.7.5.1 Configuring the Attributes Using the `ecl-include` Property

You can use the `ecl-include` property to configure attributes to be included in the ECL if an entry is modified.

Use the `dsconfig` command to set the value of the `ecl-include` property. For example, to specify that the `cn`, and `sn` attributes always be included in the ECL if an entry is modified, run the following command:

```
dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -Q -n -X \
set-external-changelog-domain-prop --provider-name "Multimaster Synchronization" \
--domain-name dc=example,dc=com \
--add ecl-include:cn --add ecl-include:sn
```

In the ECL entry that is returned by the server, the attribute name is prefixed with `target`. For example, in the previous example, the ECL entries for changes on `dc=example,dc=com` will always contain the attributes `targetcn` and `targetsn`. The values of these attributes will be the values of the `cn` and `sn` attributes of the entry before it was modified or moved.

32.7.5.2 Configuring the Attributes Using the `ecl-include-del-only` Property

In combination with the `ecl-include` property, you can use the `ecl-include-del-only` property to retrieve extra attributes for delete operations only.

Use the `dsconfig` command to set the value of the `ecl-include-del-only` property.

```
dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -Q -n -X \
set-external-changelog-domain-prop --provider-name "Multimaster Synchronization" \
--domain-name dc=example,dc=com \
--add ecl-include:cn --add ecl-include:sn --set ecl-include-del-only:true
```

32.7.6 Specifying the Attributes to be Excluded in the External Change Log

Client applications that use ECL are not always interested in all the LDAP operations executed on the server. Therefore, to avoid processing of irrelevant information you can filter a list of attributes.

You can use the `ecl-blacklist` property to configure attributes to be excluded from the ECL. It only skips MODIFY operations sent to the client application when all the modifications apply to blacklisted attributes.

> **Note:**
>
> The blacklist mechanism requires the use of the cookie mode. When you configure the blacklist property, then it prevents access to `cn=changelog` without the cookie mode.

Use the `dsconfig` command to set the value of the `ecl-blacklist` property. For example, to specify that the modify operations concerning attributes `email` and `telephonenumber` should be excluded from ECL, run the following command:

```
dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -Q -n -X \
set-external-changelog-domain-prop --provider-name "Multimaster Synchronization" \
```

```
--domain-name dc=example,dc=com --add ecl-blacklist:email \
--add ecl-blacklist:telephonenumber
```

32.7.7 Initializing Client Applications to Use the External Change Log

Client applications do not require specific server configuration to use the external change log. However, you must initialize a client application that needs to use the external change log.

- Initializing a Client Application to Use the External Change Log
- Reinitializing a Client Application When a Domain is Added
- Reinitializing a Client Application When a Domain is Removed or Disabled

> **Note:**
>
> While initializing and reinitializing client application to use ECL, you use the `lastExternalChangelogCookie` attribute. The `lastExternalChangelogCookie` attribute contains a cookie that has the required information to find the changes from the change log. Oracle Unified Directory consolidates the chronological order of ECL changes in the topology for servers that being late. However, it must be noted that in some exceptional scenarios the ECL changes are not propagated in the correct chronological order.

32.7.7.1 Initializing a Client Application to Use the External Change Log

The following example describes a scenario in which host 2 is initialised from host 1. Host 1 is not frozen during the initialization operation, so continues to receive changes. This procedure guarantees that host 2 does not lose any of the changes that were received on host 1.

1. Save the current state of host 1 by reading the last ECL cookie value on host 1.

   This is the value of the `lastExternalChangelogCookie` attribute of the root DSE. For example:

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
     -s base -b "" "objectclass=*" lastExternalChangelogCookie
     dn:
     objectClass: top
     objectClass: ds-root-dse
     lastExternalChangelogCookie: dc=europe:00000121cea5221c04b100000005 \
       00000121cea5319e04b400000009;
   ```

   Notice that host 1 is not frozen and continues to receive changes.

2. To initialize host 2, export the Oracle Unified Directory database from host 1 and import it to host2.

3. Initialize the application from the exported database.

   Restart replication on host 2, using the current state saved in Step1. The application can now start reading the ECL by providing the last cookie value as the value of the search control. For example:

   ```
   $ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -j pwd-file
     --control "1.3.6.1.4.1.26027.1.5.4:false:dc=europe:00000121cea5221c04b100000005 \
       00000121cea5319e04b400000009" -b "cn=changelog" "(objectclass=*)"
   ```

32.7.7.2 Reinitializing a Client Application When a Domain is Added

When a new replication domain is added to a topology, the ECL is enabled on that domain by default. Client applications that use the ECL must be reinitialized for the new domain.

The server enforces this requirement by rejecting SEARCH operations if the cookie that is provided does not refer to the new domain. The operation result code is `UNWILLING TO PERFORM`. The server provides a detailed message that includes a list of the domains that are missing and a cookie value for a possible partial initialization.

The client application must be reinitialized using one of the following methods:

- **Full reinitialization.** The application is reinitialized for all domains.
  1. Read the value of the `lastExternalChangelogCookie` attribute. This value refers to all domains in the topology, including the new domain.
  2. Export the database for all domains, including the new domain.
  3. Initialize the application for all domains from the export output. For more information, see Initializing a Client Application to Use the External Change Log.
  4. The application can now search the ECL using the `last_cookie_from_dse_root`.
- **Partial reinitialization.** The application is reinitialized only for the new domain.
  1. Export the database for the new domain only.
  2. Initialize the application from the export output, which contains only the entries in the new domain. For more information, see Initializing a Client Application to Use the External Change Log.
  3. The application can now search the ECL, using the cookie value for a possible partial initialization that was returned by the server in its `UNWILLING TO PERFORM` error.

> **Note:**
>
> This might result in some updates that have already been processed being replayed, because the cookie value represents the initial state of the database.

> **Note:**
>
> In draft compatibility mode, the draft API does not allow the server to enforce the application to be properly initialized. Therefore, in draft compatibility mode, any changes on the new domain are published in the ECL as soon as the new domain is added.
>
> To prevent the server from publishing changes for the new domain, follow the instructions in Disabling the External Change Log for a Specific Domain. To ensure that an application is notified of changes to a particular domain only, specify this domain either in the base DN (in cookie mode only) or as a search filter on the `targetDN` attribute.

32.7.7.3 Reinitializing a Client Application When a Domain is Removed or Disabled

When a replication domain is removed from a topology (or when the ECL is disabled for a specific domain), client applications must be alerted to the fact that no more changes will occur on that domain.

The server enforces this requirement by rejecting SEARCH operations if the cookie that is provided refers to the removed domain. The operation result code is `UNWILLING TO PERFORM`. The server provides a detailed message, that includes a list of the domains that are present in the cookie but have been removed (or for which the ECL has been disabled), and a cookie value for a possible continuation.

The client application can use one of the following methods to handle the removed domain:

- **Smooth continuation.** In this case, the application applies its own policy of what to do when a domain is removed. To assist with the formulation of this policy, the application can search the ECL by providing the cookie value for a possible continuation that is returned by the server in the error message.
- **Full reinitialization.** The application is reinitialized for all domains.
  1. Read the value of the `lastExternalChangelogCookie` attribute. This value refers to all domains in the topology, excluding the removed domain.
  2. Export the database for all domains.
  3. Initialize the application for all domains from the export output. For more information, see Initializing a Client Application to Use the External Change Log.
  4. The application can now search the ECL using the `lastExternalChangelogCookie`.

32.7.8 Controlling Access to the External Change Log

Access to the ECL is ruled by global ACIs, which you can configure on the server. By default, only the root user can access the ECL.

For information about configuring global ACIs, see Managing Global ACIs Using `dsconfig`.

32.7.9 Purging the External Change Log

The External Change Log is purged simultaneously with the replication change log.

For information about changing the interval at which the replication change log is purged, see Configuring Replication Purge Delay.

Sometimes, an application might submit a search request on the ECL, providing a cookie value that is older than the oldest change stored on the server (because a purge has occurred since the last request from that application). In this case, the server rejects the requests and indicates that the cookie is too old and that a full resync is required.

32.7.10 Disabling the External Change Log on a Server

Use the `dsreplication disable-changelog` command to disable the external change log on a server, for a specific base DN.

```
$ dsreplication disable-changelog -h localhost -p 4444 -D "cn=directory manager" \
  -j pwd-file -b dc=example,dc=com -X -n
```

32.7.11 Disabling the External Change Log for a Specific Domain

In certain situations, you might want to exclude changes on a specific domain from the external change log. You can disable the ECL for a specific replication domain, which prevents changes to this domain from being published in the ECL.

1. Obtain the domain name, as described in Retrieving the Replication Domain Name.
2. Set the external changelog domain properties for that domain.

   For example, to prevent changes to the schema from being published in the ECL, run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  set-external-changelog-domain-prop \
  --provider-name "Multimaster Synchronization" --domain-name cn=schema \
  --set enabled:false
```

32.7.12 Retrieving the Last Change Number

Use the `ldapsearch` command to get the last change number attribute on the server.

> **Note:**
>
> Ensure that the external change log is enabled, as described in Enabling the External Change Log.

Run this command:

```
./ldapsearch -h <hostname> -p <portnumber> -D "cn=Directory Manager"
-w <password> -s base -b "" objectclass=* lastchangenumber
```

Example

```
$ ldapsearch -h localhost -p 1389 -D "cn=directory manager" -w <password>
  -s base -b "" "objectclass=*" lastchangenumber
```

32.7.13 Porting Applications that Rely on Other Change Logs

The external change log is based on the LDAP change log draft but does not strictly support this change log. The LDAP change log draft uses an integer as the key to browse the change log whereas the external change log uses a cookie.

For more information on the LDAP change log draft, see http://tools.ietf.org/html/draft-good-ldap-changelog-04.

On the client side, the cookie mechanism has the following advantages:

- Ability to fail-over from one ECL instance to another
- Ability to load balance request over several ECL instances

On the server side, the cookie mechanism has the following advantages:

- Easier implementation in a multi-master environment
- Cheaper in terms of resources required on the server

- Smaller performance impact for other applications that generate changes

> **Note:**
>
> The Oracle Directory Server Enterprise Edition (ODSEE) Retro Change Log (RCL) supports the LDAP change log draft, with some specific additions.

This section describes the following topics:

- Understanding the Differences Between the ECL and the LDAP Change Log Draft.
- Understanding the Differences Between the ECL and the Oracle Directory Server Enterprise Edition Retro Change Log.
- About the API for Compatibility With the LDAP Change Log Draft and the Oracle Directory Server Enterprise Edition Retro Change Log.

32.7.13.1 Understanding the Differences Between the ECL and the LDAP Change Log Draft

The following topics describe the differences between the two change logs, which will assist you in porting client applications:

- About Index Differences.
- About DIT and Schema Differences.

32.7.13.1.1 About Index Differences

The LDAP change log draft specifies the change log index as an integer (`changenumber` attribute). This works well when the change log is served by a single server (which was the case at the time that the LDAP change log draft specification was written.) When the change log service supports more than one server and when failover is supported from one server to another, the integer format is not appropriate.

> **Note:**
>
> You should index the `replicationCSN` attribute on `cn=changelog` for compatibility with Oracle Directory Server Enterprise Edition. If you index the `replicationCSN` attribute on parameters other than `cn=changelog`, the index might have a performance impact.

32.7.13.1.2 About DIT and Schema Differences

The LDAP change log draft specifies the DN for entries in the change log as `changenumber=changenumber,cn=changelog`. The ECL uses the following DN for entries in the change log:

`replicationcsn=`*`replicationCSN`*`,`*`replication-domain-DN`*`,cn=changelog`

The ECL schema is based on the LDAP change log draft schema, however, Oracle Unified Directory manages an index in the ECL through a cookie that is opaque to the application, rather than through the `changenumber` attribute.

The following table describes the schema differences:

Origin	MUST	MAY

LDAP Change Log Draft	`changenumber`
	`targetDn`
	`changetype`
		`changes`
		`newRDN`
		`deleteOldRDN`
		`newSuperior`

ODSEE RCL	`changenumber`
	`targetDn`
	`changetype`
	`changetime`
		`changes`
		`newRDN`
		`deleteOldRDN`
		`newSuperior`
		`changeHasReplFixupOp`
		`changeIsReplFixupOp`
		`deletedEntryAttrs`
		`replicaIdentifier` (operational)
		`replicationCSN` (operational)
		`targetUniqueId` (operational)

Oracle Unified Directory ECL	`changenumber`
	`targetDn`
	`changetype`
	`changetime`
		`changes`
		`newRDN`
		`deleteOldRDN`
		`newSuperior`
		`replicaIdentifier` (operational)
		`replicationCSN` (operational)
		`targetentryuuid` (operational)
		`changelogcookie` (operational)

32.7.13.2 Understanding the Differences Between the ECL and the Oracle Directory Server Enterprise Edition Retro Change Log

Schema and implementation-based values
The Oracle Directory Server Enterprise Edition RCL specifies that the target entry unique ID is stored in the `targetuniqueid` attribute. The format of this attribute value is specific to Oracle Directory Server Enterprise Edition. The `replicationcsn` attribute also has a value that is specific to Oracle Directory Server Enterprise Edition.

First and last ECL index
The Oracle Directory Server Enterprise Edition RCL supports the following attributes in the root DSE entry:

- The `firstchangenumber` attribute, which contains the first (oldest) change log index as an integer change number.

This value is updated when the change log is purged. Before connecting to the change log server, an application reads the first change log index and compares it with the change log index that it stored. If the first change log index is more recent than the last change log index stored by the application, the application knows that the changes from the application index to the first change log index will never be returned by the server. They can only be obtained by reading the entries (full resync).

With the Oracle Unified Directory ECL, this procedure is not required of the application. Instead the Oracle Unified Directory server does the check and rejects the request when the cookie is too old. For more information, see Using the External Change Log.

- The `lastchangenumber` attribute, which contains the latest (newest) change log index as an integer change number.

  The Oracle Unified Directory ECL supports the equivalent feature with the `lastExternalChangelogCookie` attribute. The `lastExternalChangelogCookie` attribute contains a cookie that has the required information to find the changes from the change log. For more information, see Using the External Change Log.

Purge delay
In the Oracle Directory Server Enterprise Edition RCL, the external change log and the regular replication change log are different databases. In Oracle Unified Directory, the two change logs are in the same database. This design decision has several advantages. An additional consequence of this design decision is that Oracle Directory Server Enterprise Edition can have two different trim policies (purge delays), while in Oracle Unified Directory the trim policy is the same.

32.7.13.3 About the API for Compatibility With the LDAP Change Log Draft and the Oracle Directory Server Enterprise Edition Retro Change Log

Oracle Unified Directory provides an additional API that is compatible with the LDAP draft change log and supports most of the additional features of the Oracle Directory Server Enterprise Edition Retro Change Log. The use of this API has a performance impact in terms of CPU and database (disk) space on the server side, and some computation for the application that fails over from one ECL server to another one.

The use of this compatible API (*compatible mode*) is configured when the server receives a request on the ECL with no change log cookie. The server returns entries with a `changenumber` attribute, the value of which is an incremental integer.

The client can search the ECL by providing a filter on the `changenumber`. The target entry unique ID is stored in an attribute called `targetuniqueid` with a format compatible with the Oracle Directory Server Enterprise Edition Retro Change Log. The first and last `changenumber` are present as attributes of the root DSE entry.

Limitations of the Compatibility API

Because Oracle Unified Directory does not store the ECL in a dedicated database, it does not support all the features supported by a JEB back end, such as specific indexes.

In addition, to support the `changenumber`-based ordering that is specified by the LDAP change log draft, Oracle Unified Directorymust store a mapping from the `changenumber` to the replication state. When the server processes a request, it must try to retrieve the replication state from the `changenumber` that is provided in the request filter. If this cannot be achieved, the request is rejected.

32.8 Managing Tombstones in Oracle Unified Directory

Oracle Unified Directory supports the tombstone feature to maintain those directory entries that are deleted on one replica until they are no longer needed for replication.

This section contains the following topics:

32.8.1 About Tombstone Support

When an entry is a target of an LDAP delete operation, Oracle Unified Directory normally deletes this entry from the directory database. However, when the tombstone creation is enabled, that delete operation is a logical delete, which means the directory does not physically remove that entry from the database. Instead, the directory converts the entry into a tombstone entry with a specific object class. Tombstone entries use their `nsUniqueID` as RDN.

Tombstones provide administrators with the ability to resurrect one or more deleted entries that were accidentally deleted to the original entries, if required.

Tombstone support helps the Oracle Unified Directory replication solves some deletion conflicts. During replication, a server may crash due to a connection failure between the Directory Server and the Replication Server. After the crash, it is possible that some operations were committed in the database of the Directory Server but not yet transmitted to a Replication Server. In such cases, Replication Servers use tombstone entries internally to resolve conflicts.

> **Note:**
>
> Problems may arise if a replica (say A) with the tombstones may be offline beyond the purge interval. When this replication is connected to the replication ring, it may generate conflicts with other servers where these tombstone entries may have been purged. It is best to initialize A with the chosen master from the replication topology.

32.8.2 About Tombstone Entries

A tombstone is a read-only entry and is stored with a different DN. However, from a client point of view, the tombstone keeps the same DN as the deleted entry.

Tombstone entries have special object class values; either `tombstone` or `nstombstone`. The tombstone entry can contain all the attributes of the original deleted entry or only a part of them.

```
 dn: cn=u2,cn=users,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
```

```
objectClass: orclIDXPerson
objectClass: nstombstone
objectClass: top
objectClass: tombstone
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: orclIDXPerson
objectClass: top
objectClass: tombstone
objectClass: nstombstone
givenName: useri
description: bidule
cn: usersSuffix
cn: u2
sn: u2
userPassword: {SSHA}zL22oCGjlG80cmbvh9jnzXAIyUfDSO+y4gRi+w==
l: London
orclGUID: 859b759ea60746a0bb271e4a46007f24
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
deleteTimestamp: 20110922084753Z
subschemaSubentry: cn=schema
proximity: -1
createTimestamp: 20110922084752Z
pwdChangedTime: 20110922084752.126Z
structuralObjectClass: orclIDXPerson
entryDN: cn=u2,cn=users,dc=example,dc=com
entryUUID: 859b759e-a607-46a0-bb27-1e4a46007f24
creatorsName: cn=Internal Client,cn=Root DNs,cn=config
modifyTimestamp: 20110922084753Z
nscpEntryDN: cn=u2,cn=users,dc=example,dc=com
modifiersName: cn=Internal Client,cn=Root DNs,cn=config
 1316681273304
```

32.8.3 Enabling or Disabling Tombstone Support

The tombstone support is enabled by default in Oracle Unified Directory. You can disable or enable tombstones using the `tombstone-creation-enabled` advanced property of the DB Local Backend Workflow Element.

To disable tombstone for the `userRoot` backend:

```
dsconfig set-workflow-element-prop \
          --element-name userRoot \
          --set tombstone-creation-enabled:false \
          --hostname localhost \
          --port 1444 \
          --trustAll \
          --bindDN cn=directory\ manager \
          --bindPasswordFile /local/tests/password \
          --no-prompt
```

For the parameter description of the *tombstone-creation-enabled* advanced property, see *DB Local Backend Workflow Element* in *Configuration Reference for Oracle Unified Directory*.

32.8.4 Searching for Tombstone Entries

Tombstone entries do not show up in regular search operations, unless you add `objectclass=tombstone` or `objectclass=nstombstone` in your search filter in the search request.

The following `ldapsearch` command returns tombstone entries under `dc=example,dc=com`:

```
$ ldapsearch -h localhost -p 5444 -D "cn=Directory Manager" -j pwd-file.txt -b
dc=example,dc=com "(objectclass=nsTombstone)"
dn: uid=user.3,ou=people,dc=com
postalAddress: Aaron Atrc$59748 Willow Street$Green Bay, TN 66239
postalCode: 66239
description: This is the description for Aaron Atrc.
uid: user.3
userPassword: {SSHA512}li0gmRHdPtL326Pc2B2tIfGe/RdITcQXzZshsR96nKl25FaTYFXvp9nq1
rNzafjKKFkgRrhGEwDggn6KaMsjym0Ggzm36oAh
employeeNumber: 3
initials: AKA
givenName: Aaron
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: nstombstone
objectClass: top
objectClass: tombstone
pager: +1 197 025 3730
mobile: +1 890 430 9077
cn: Aaron Atrc
telephoneNumber: +1 094 100 7524
sn: Atrc
street: 59748 Willow Street
homePhone: +1 332 432 4295
mail: user.3@maildomain.net
l: Green Bay
st: TN
orclGUID: 8264e7f807f438879b9833ca1d0ed04a
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
deleteTimestamp: 20150227113328Z
subschemaSubentry: cn=schema
changelog: cn=changelog
structuralObjectClass: inetOrgPerson
dssynchist:
dn:0000014bcad0132714a000000003:del
nsUniqueId: 8264e7f807f438879b9833ca1d0ed04a
entryDN: uid=user.3,ou=people,dc=com
entryuuid: 8264e7f807f438879b9833ca1d0ed04a
nscpEntryDN: uid=user.3,ou=people,dc=com
modifyTimestamp: 20150227113328Z
modifiersName: cn=directory manager
```

32.8.5 Purging Tombstone Entries Automatically

Tombstone entries consume resources (index, database storage) and hence are not stored indefinitely. The purge thread is invoked periodically at a specified interval as per the configuration parameter `tombstone-purge-interval`.

The tombstone purge thread deletes tombstone entries that are older than the time specified in the `tombstone-lifetime` configuration parameter. Tombstone entries have a lifetime of one week set by default. After this delay, the server automatically cleans these entries.

> **Note:**
>
> Ensure that the time interval of the `tombstone-lifetime` parameter is not too low. An aggressive tombstone lifetime interval may make the server not responsive, if you have a lot of tombstones to purge.

To change the tombstone lifetime to a particular interval:

```
dsconfig set-workflow-element-prop \
          --element-name userRoot \
          --set tombstone-lifetime:1440\ m \
          --hostname localhost \
          --port 1444 \
          --trustAll \
          --bindDN cn=directory\ manager \
          --bindPasswordFile /local/tests/password \
          --no-prompt
```

To change the tombstone purge interval:

```
dsconfig set-workflow-element-prop \
          --element-name userRoot \
          --set tombstone-purge-interval:6\ m \
          --hostname localhost \
          --port 1444 \
          --trustAll \
          --bindDN cn=directory\ manager \
          --bindPasswordFile /local/tests/password \
          --no-prompt
```

For the parameter description of the *tombstone-lifetime* and *tombstone-purge-interval* properties, see *DB Local Backend Workflow Element* in *Configuration Reference for Oracle Unified Directory*.

32.8.6 Removing Tombstone Entries

You can also remove tombstone entries without waiting for an automatic purge.

To remove tombstones entries:

1. Export a data set from one server, excluding tombstone entries from the export with a search filter:

   ```
   /export-ldif -n userRoot -l /tmp/export_no_tombstones.ldif -excludeFilter
   "objectclass=tombstone"
   ```

This filter excludes every entry having the `objectclass=tombstone` from the LDIF file.

2. Stop the directory server where you need to import the file including the one where you performed the `export-ldif`.
3. Import the LDIF file:

   ```
   /import-ldif -n userRoot -l /tmp/export_no_tombstones.ldif
   ```

4. Start the server again.

You can perform these steps for each and every directory server in the topology to remove the tombstone entries. The `dsreplication status` command shows no difference among the back-end entry counts.

32.9 Configuring Schema Replication

Schema replication is enabled by default. When you configure replication as part of the server setup, the schema of the new server is automatically initialized with the schema of the existing server in the topology.

This section covers the following topics:

- Specifying the Schema Source
- Disabling Schema Replication

32.9.1 Specifying the Schema Source

When you configure replication with the `dsreplication enable` command, you can specify that the schema of the second directory server be used to initialize the schema of the first server. If you do not specify an option, the schema of the first directory server is used by default.

In the following example, the data of `host1` is used to initialize `host2` but the schema of `host2` is used to initialize the schema on `host1`:

```
$ dsreplication enable --host1 host1 --port1 4444 \
  --bindDN1 "cn=Directory Manager" --bindPasswordFile1 pwd.txt \
  --replicationPort1 8989 --host2 host2 --port2 4444 \
  --bindDN2 "cn=Directory Manager" --bindPasswordFile2 pwd.txt \
  --replicationPort2 8989 --adminUID admin --adminPasswordFile pwd.txt \
  --baseDN "dc=example,dc=com" --useSecondServerAsSchemaSource -X
```

32.9.2 Disabling Schema Replication

In certain circumstances, you might not want the schema to be replicated. The schema is replicated under a separate base DN, `"cn=schema"`.

This section covers the following topics:

- Specifying That Schema Should Not Be Replicated.
- Disabling Schema Replication.

32.9.2.1 Specifying That Schema Should Not Be Replicated

When you configure replication with the `dsreplication enable` command, you can specify that the schema should not be replicated, using the `--noSchemaReplication` option.

> **Note:**
>
> If you use QuickSetup to enable replication, you cannot specify that the schema should not be replicated.

32.9.2.2 Disabling Schema Replication

In an existing topology in which the schema are being replicated, you can disable this functionality by disabling replication of the schema base DN. The following example disables schema replication from the directory server running on the local host on port `1389`:

```
$ dsreplication disable -h localhost -p 1389 -D "cn=directory manager" \
  -j pwd-file -b "cn=schema" -X
```

> **Note:**
>
> The previous example does not disable schema replication for the entire topology. To disable schema replication for the entire topology, you must run the equivalent command for each directory server in the topology.

32.10 Replicating to a Read-Only Server

The Oracle Unified Directory replication model is a multi-master model, that is, all the replication servers in the topology can process both read and write operations. However, you can configure a directory server to be read-only, in which case add, modify, and delete operations from LDAP clients are rejected on this server.

> **Note:**
>
> A read-only directory server functions like a *consumer replica* does in the Oracle Directory Server Enterprise Edition replication model.

This example assumes a replication configuration with replication servers on two hosts, host1 and host2. The example makes the directory server on host2 a read-only replica. The example uses the `dsconfig` command, which accesses the server configuration through the administration connector. See Managing Administration Traffic to the Server.

Use the `dsconfig` command to set the `writability-mode` of host2.

```
$ dsconfig -h host2 -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-global-configuration-prop --set writability-mode:internal-only
```

A writability mode of *internal-only* means that replication operations are processed on the server, but the server is not writeable directly by LDAP client applications.

32.11 Detecting and Resolving Replication Inconsistencies

Directory server replication has been designed to ensure that replicated databases remain consistent, even for hardware faults, directory server restarts, or network failures. However, despite these efforts, possible hardware failures (disk errors, memory errors) or software errors (causing memory corruption) might lead to inconsistent databases.

The following topics explain how to detect replication inconsistencies, and how to resolve them when they are identified:

- About the Types of Replication Inconsistencies.
- Detecting Inconsistencies.
- Resolving Inconsistencies.
- Solving Naming Conflicts.

32.11.1 About the Types of Replication Inconsistencies

When inconsistencies occur, they might remain hidden for some time or they might trigger replication or application errors.

Examples of inconsistencies include the following:

- An entry is present on all but one directory server in the replication topology.
- An entry has a DN on one directory server that is different to its DN on all other directory servers.
- An entry has different attributes on one directory server than on other directory servers in the replication topology.

32.11.2 Detecting Inconsistencies

You can detect inconsistencies during replication by checking the replication log file and the errors reported by users.

- **Check for information in the replication log file**. The replication log file is configured by default and lists inconsistencies that are detected by the replication mechanism. Imagine, for example, that a modify operation is performed on an entry that is missing from one directory server in the topology. When replication attempts to replay this operation to that server, it will detect the problem and produce an error in the `logs/replication` error log. This kind of error will not stop replication, but the operation will not be replayed and the administrator will need to repair the inconsistency.
- **Pay attention to errors reported by client applications or users**. Client applications or users might experience errors when accessing the directory server that might be due to replication inconsistencies.
- **Make regular checks for database consistency**. With the current directory server release, these checks must be performed manually, using searches or database exports.

32.11.3 Resolving Inconsistencies

If a replication inconsistency is found on a single directory server in the topology, it is not possible to fix this inconsistency using regular LDAP operations. This is because the LDAP operation itself would be replicated to the other directory servers in the topology and might

cause damage on those servers. In addition, the fix might involve modifying attributes that are generated by the directory server, such as the `entryuid` or `modifyTimestamp` attributes. Regular LDAP operations cannot modify such attributes.

Replication repair operations must therefore be done using LDAP operations that specify the Replication Repair Control (OID: `1.3.6.1.4.1.26027.1.5.2`).

> **Caution:**
>
> Because the replication repair control allows you to skip several controls usually done by the directory server, it should be used with great care and only when consistency problems have been detected and asserted.

The repair control alters the regular processing of an operation as follows:

- The operation can modify attributes that might not normally be modified or added (`NO-USER-MODIFICATION`), such as `entryuuid` and `ds-sync-hist`.
- No replication change number is associated with the operation.
- The operation is not published to the replication server and is therefore a local-only operation.
- Replication does not try to resolve conflicts or to generate historical information for this operation.
- Most of the schema checks are not performed for this operation.

For example, the following `ldapmodify` operation repairs an entry on `host1` only, with the changes contained in the file `changes.ldif`:

```
$ ldapmodify -J 1.3.6.1.4.1.26027.1.5.2 -h localhost -p 1389 \
  -D "cn=Directory Manager" -j pwd-file -f changes.ldif
```

When you repair an entry, you must repair all of its regular attributes as well as the attributes generated by the directory server, such as `modifyTimestamp`, `modifiersName`, `createTimestamp`, `creatorsName`, and `ds-sync-hist`. The values of these attributes should be read from a directory server that contains the correct values, and recreated on the server with faulty values.

The `ds-sync-hist` attribute contains historical information that replication uses to solve modify conflicts. This attribute can only be viewed by an administrator.

32.11.4 Solving Naming Conflicts

Entries with identical DNs can be created on separate directory servers if they are created before the servers replicate the changes to each other. When the remote operation is replicated to the local server, a naming conflict occurs. The naming conflict results in the creation of a *conflict entry* on the local server.

Conflict entries have a specific DN, of the form `entryuuid=entryUid+oldRDN`. Every conflict entry includes a `ds-sync-conflict` attribute, whose value is the DN of the conflicting regular entry.

For example, imagine that the entry `cn=bjensen,ou=People,dc=example,dc=com` is created simultaneously on two directory servers. The entry on server 1 is given a unique ID of uid1 and

the entry on server 2 is given a unique ID of uid2. Both directory servers will have the following two entries after replication:

```
cn=bjensen,dc=example,dc=com
...
entryuuid=uid2+cn=bjensen,dc=example,dc=com
  ds-sync-conflict:cn=bjensen,dc=example,dc=com
```

When you have identified the conflicting entry, you can rename it so that it has a unique DN.

If the naming attribute in a conflicting entry is multi-valued, you can rename the conflicting entry as follows:

1. Rename the entry while keeping the old RDN value, for example:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: entryuuid=uid2+cn=bjensen,dc=example,dc=com
   changetype: modrdn
   newrdn: cn=bljensen
   deleteoldrdn: 0
   ^D
   ```

   You cannot delete the old RDN value in this step because it also contains the `entryuuid` operational attribute, which cannot be deleted.

2. Remove the old RDN value of the naming attribute and the conflict marker attribute, for example:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: cn=bljensen,dc=example,dc=com
   changetype: modify
   delete: cn
   cn: bjensen
   delete: ds-sync-conflict
   ^D
   ```

If the naming attribute in a conflicting entry is single-valued, for example `dc` (domain component), you cannot simply rename the entry to another value of the same attribute. Instead, you must give the entry a temporary name, as follows:

1. Rename the entry by using a different naming attribute, and keep the old RDN, for example:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: entryuuid=uid2+dc=HR,dc=example,dc=com
   changetype: modrdn
   newrdn: o=TempHR
   deleteoldrdn: 0
   ^D
   ```

   You cannot delete the old RDN value in this step because it also contains the `entryuuid` operational attribute, which cannot be deleted.

2. Change the desired naming attribute to a unique value and remove the conflict marker attribute, for example:

   ```
   $ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
   dn: o=TempHR,dc=example,dc=com
   changetype: modify
   replace: dc
   dc: NewHR
   delete: ds-sync-conflict
   ^D
   ```

3. Rename the entry back to the intended naming attribute and delete the temporary RDN, for example:

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file
dn: dc=NewHR,dc=example,dc=com
changetype: modrdn
newrdn: dc=NewHR
deleteoldrdn: 1
^D
```

32.12 Managing Certificates Using `dsreplication`

The replicated Oracle Unified Directory servers use certificates to perform authentication and to encrypt the replication communication.

You can manage these certificates using the following `dsreplication` subcommands:

- Listing Certificates Using `dsreplication list-certs`
- Regenerating Certificates Using `dsreplication regenerate-cert`
- Providing Certificates Using `dsreplication set-cert`
- Verifying and Fixing Certificates Using `dsreplication verify`

> **Note:**
>
> The certificates described in this section are used in the internal replication communication process and are different from certificates used by the server to communicate with LDAP clients.

For more information about the `dsreplication` subcommands, including the syntax, see dsreplication.

32.12.1 Listing Certificates Using `dsreplication list-certs`

The `list-certs` subcommand displays the certificates used by the replicated servers in a deployment.

For example, to display all certificates in a replication deployment:

```
$ dsreplication list-certs -j /tmp/password.txt -X -n
Establishing connections ..... Done.

Reading Certificates ..... Done.

host1.example.com:4444
==============
User DN:   CN=host1.example.com, O=Oracle Unified Directory Certificate
Validity:  From November 7, 2013 1:48:55 PM CET to November 2, 2033 1:48:55 PM CET
Issuer:    CN=host1.example.com, O=Oracle Unified Directory Certificate

host1.example.com:5444
==============
User DN:   CN=host1.example.com, O=Oracle Unified Directory Certificate
Validity:  From November 7, 2013 1:48:55 PM CET to November 2, 2033 1:48:55 PM CET
Issuer:    CN=host1.example.com, O=Oracle Unified Directory Certificate
```

32.12.2 Regenerating Certificates Using `dsreplication regenerate-cert`

The `regenerate-cert` subcommand regenerates the Certificate used by the specified server, or for all servers, in the replication topology. By default, the Oracle Unified Directory server automatically generates some Certificates for replication. This command allows you to regenerate these Certificates if needed (for example, because they are about to expire).

> **Note:**
>
> If you provided your own Certificates (for example, using the `dsreplication set-cert` subcommand), Oracle does not recommend that you use this command because it will try to remove those Certificates.

For example, to regenerate the Certificate of the `host1.example.com` server using port 4444:

```
$ dsreplication regenerate-cert -h host1.example.com -p 4444 -X -j /tmp/password.txt -n
--adminUID admin
Establishing connections ..... Done.

After the generation of the new certificate for server host1.example.com:4444,
the references to the old certificate will be removed.

Regenerating the Certificate of Server host1.example.com:4444 ..... Done.
Propagating certificate public keys ..... Done.
Reestablishing replication connections on server host1.example.com:4444 .... Done.
Checking registration information ..... Done.
```

To regenerate the Certificates of all the servers that are replicated, include the `--all` option.

In addition, if the data-encryption is also enabled and there are entries with attributes encrypted and stored, you should use the `regenerate-cert` command to renew the Self-Signed Certificates. The `regenerate-cert` command automatically resets the Certificates for the server in the replication topology and synchronizes the admin data. You need not export and re-import the data required for attribute encryption. Further, you need not restart the server.

> **Note:**
>
> You should not use the Keytool commands to update the keystore file directly.

32.12.3 Providing Certificates Using `dsreplication set-cert`

The `set-cert` subcommand allows you to provide the Certificate that the replication system should use. You also provide the keystore containing the public key to be used to communicate with the other replicated servers.

For example, after you generate a self-signed Certificate named `my-ads-keystore` using a utility such as `keytool`, invoke the `set-cert` subcommand, as follows:

```
$ dsreplication set-cert -X

>>>> Specify Oracle Unified Directory LDAP connection parameters
```

```
Directory server hostname or IP address [host1.example.com]:
Directory server administration port number [4444]:
Global Administrator User ID [admin]:
Password for user 'admin':

Establishing connections ..... Done.

Choose the type of the key store.
    1)  JKS
    2)  JCEKS
    3)  PKCS12
    4)  PKCS11
    5)  Other (File Based)
    6)  Other (Hardware Based)
    q)  quit

Enter choice [1]:

You must provide the path of the key store containing the certificate to be
used by the replication.  The server must have read access rights to thispath.

Key store path: /users/admin/my-ads-truststore

You must provide the path of the file containing the password (PIN) in clear
of the key store.  The server must have read access rights to this file.

Key store password (PIN) file: /tmp/password.txt

The server allows to encrypt the key store password file '/tmp/password.txt'.
Note that the server must have write access rights on the file to do so.
Do you want to encrypt the key store password file? (yes / no) [no]:

Choose the Nickname of the Certificate:
    1)  my-ads-truststore

    q)  quit
Enter choice [my-ads-truststore]:
Updating the certificate configuration of server host1.example.com:4444 ..... Done.
Propagating certificate public keys ..... Done.
Reestablishing replication connections on server
host1.example.com:4444 ........................... Done.
Checking registration information ..... Done.
See /tmp/oud-replication-356794289708010450.log
for a detailed log of this operation.
```

In addition, if data-encryption is also enabled and there are entries with attributes encrypted and stored, you should use the `set-cert` command to renew the Custom or CA Certificates (for example, when the Certificates are about to expire).

You provide the path to the new keystore which consists of the new private-public key pair, or you can provide the same keystore path if you are only renewing the public Certificate. This automatically resets the Certificates for the server and its references in the replication topology and synchronizes the admin data. You need not export and re-import the data required for attribute encryption. Further, you need not restart the server.

32.12.4 Verifying and Fixing Certificates Using `dsreplication verify`

The `verify` subcommand allows you to verify the configuration of the replicated servers and then (in interactive mode) to fix any problems related to the certificates used by the replication system, if needed.

To verify and correct certificates used by the replication system, run the `verify` subcommand in interactive mode. For example:

```
$ dsreplication -X

What do you want to do?

    1)   Enable Replication
    2)   Disable Replication
    3)   Initialize Replication on one Server
    4)   Initialize All Servers
    5)   Pre External Initialization
    6)   Post External Initialization
    7)   Display Replication Status
    8)   Purge Historical
    9)   Set the Trust Flag of a Directory Server
    10)  Enable External Changelog
    11)  Disable External Changelog
    12)  Verify Server Configuration
    13)  >>>> Replication Certificate Management

    q)   quit

Enter choice: 12

>>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server host name or IP address [host1.example.com]:

Directory server administration port number [4444]:

Global Administrator User ID [admin]:

Password for user 'admin':

Establishing connections ..... Done.

No errors were found with the configured host names.  The following host names
have been found in the registration information to identify the different
replicated servers:
host1.example.com
host2.example.com
host3.example.com

Do you want to update the host names for the servers? (yes / no) [no]: no

The replication servers are consistently referenced in the configuration.
The replication server values in the configuration are:
- host1.example.com:8989
- host2.example.com:8989
- host3.example.com:9989

What do you want do?
```

```
    1)  Provide directly the replication server values to be used
    2)  Do not update the configuration

Enter choice [2]:

Checking certificates ..... Done.

The following certificates are missing in the trust store of server
host1.example.com:4444:
- Certificate of Server host2.example.com:4444

Do you want to repair the issues with the certificates? (yes / no) [yes]: yes
Fixing certificates ..... Done.
Reestablishing replication connections on server host1.example.com:4444 .......... Done.
Reestablishing replication connections on server host2.example.com:4444 .......... Done.
Reestablishing replication connections on server host3.example.com:4444 .......... Done.
Checking registration information ..... Done.

See /tmp/oud-replication-356794289708010450.log
for a detailed log of this operation.
```

32.13 Using `verify` Subcommand

Review these topics for a contextual description of `verify` subcommand and an example how to use the `verify` subcommand with `dsreplication`.

- About `verify` Subcommand.
- Verifying and Fixing a Replication Configuration Using `dsreplication verify`.

32.13.1 About `verify` Subcommand

The `verify` subcommand allows you to verify the replication configuration of the replicated servers and then (in interactive mode) fix any inconsistencies, if needed.

Oracle recommends that you run the `verify` subcommand in interactive mode (that is, without the `--no-prompt` option). If any inconsistencies are found in the replication configuration, they will be displayed and you can fix them interactively.

Use the `verify` subcommand to:

- Remove references to servers that are no longer reachable (for example, because they crashed and are not recoverable or they were not properly uninstalled).
- Fix configuration problems related to the certificates used by the replication system.
- Update the host names used by the replication configuration.

32.13.2 Verifying and Fixing a Replication Configuration Using `dsreplication verify`

Run the `verify` subcommand in interactive mode to verify and fix a replication configuration.

For example:

```
$ dsreplication -X
What do you want to do?

    1)   Enable Replication
```

```
    2)   Disable Replication
    3)   Initialize Replication on one Server
    4)   Initialize All Servers
    5)   Pre External Initialization
    6)   Post External Initialization
    7)   Display Replication Status
    8)   Purge Historical
    9)   Set the Trust Flag of a Directory Server
    10)  Enable External Changelog
    11)  Disable External Changelog
    12)  Verify Server Configuration
    13)  >>>> Replication Certificate Management

    q)   quit

Enter choice: 12


>>>> Specify Oracle Unified Directory LDAP connection parameters

Directory server host name or IP address [host1.example.com]:

Directory server administration port number [4444]:

Global Administrator User ID [admin]:

Password for user 'admin':

Establishing connections ..... Done.

No errors were found with the configured host names.  The following host names
have been found in the registration information to identify the different
replicated servers:
host1.example.com
host2.example.com
host3.example.com

Do you want to update the host names for the servers? (yes / no) [no]:

The following replication servers do not have the complete list of replication
server values in their configuration:
- host2.example.com:8989
The replication servers must have the complete list of replication server
values.

The following replication domains do not have the complete list of replication
server values in their configuration:
- host2.example.com:4444(cn=admin data)
- host2.example.com:4444(cn=schema)
- host2.example.com:4444(dc=example,dc=com)
If they have not been configured this way intentionally, the configuration of
the replication domains should be updated.

The replication server values in the configuration are:
- host1.example.com:8989
- host2.example.com:8989
- host3.example.com:8989

What do you want do?

    1)  Use the interactive assistant
    2)  Provide directly the replication server values to be used
```

```
    3)  Do not update the configuration

Enter choice [1]:

The replication server values proposed after running the assistant are:
- host1.example.com:8989
- host2.example.com:8989
- host3.example.com:8989

What do you want to do?

    1)  Use the values above
    2)  Use the values above but do not update the replication domains
    3)  Provide the values again
    4)  Do not update the configuration

    q)  quit

Enter choice [1]:

Checking certificates ..... Done.
No problems were found with the certificates used by the replication.

Updating replication server references ..... Done.

See /tmp/oud-replication-6260669521027550543.log
for a detailed log of this operation.
```

32.14 Understanding Purging Historical Replication Data

Oracle Unified Directory maintains a history of all changes that have been made on the server as a result of replication operations. This historical replication data is stored in an attribute of each user entry, and can eventually take up a large amount of space on your disk. Historical information is therefore purged when an entry is modified, or when you specifically run a command to purge the data.

By default, information that is older than one day is purged. You can specify the age of data that should be purged by setting the value of the `conflicts-historical-purge-delay` property of the replication domain. The following example specifies that data older than five days should be purged. The property value is expressed in minutes.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-replication-domain-prop --provider-name "Multimaster Synchronization" \
  --domain-name dc=example,dc=com --set conflicts-historical-purge-delay:7200m
```

You can also purge historical data immediately, or schedule a task to purge the data at a specific time. Imagine, for example, that you initialize a server with a large number of entries, then perform a significant number of changes on these entries. The resulting replication historical data will increase the size of the database quite substantially. If your server is then used mainly for read operations, the large database size remains, because no modifications are made to trigger a purge of the historical data. In this case, you can launch a once off purge task to remove the historical data that was generated by the initial modifications, and return the database to a more accurate size.

Because the purge process can take some time, you are required to specify the maximum duration of the purge (in seconds). To purge historical data immediately, run the following command:

```
$ dsreplication -h localhost -p 4444 --adminUID admin --adminPasswordFile pwd.txt \
  purge-historical --maximumDuration 3600 --baseDN dc=example,dc=com -X -n
```

For information about scheduling commands as tasks, see Configuring Commands As Tasks.

32.15 Understanding Isolated Replicas

Review these topics to understand isolated replicas and their deployment scenarios.

This section covers the following topics:

- About Isolated Replicas.
- Understanding the Deployment Scenarios for Isolated Replicas.

32.15.1 About Isolated Replicas

An *isolated replica* is a directory server that can accept changes from other replicas for replay but cannot send changes to the replication server to which it is connected. An isolated replica cannot be the source of data updates to the topology. You can use isolated replicas to separate a directory server from the rest of the replication topology.

Every directory server in the topology has a `trusted` configuration property that is set to `true` by default. Isolated replicas are identified as such by configuring them as *untrusted* servers in the topology, that is, by setting the `trusted` configuration property to `false`. Data that comes from an untrusted directory server is discarded by a replication server. This ensures that an isolated replica cannot be the source of data updates in the replication topology.

Only *directory servers* are configured as trusted or untrusted. Replication servers do not have the trusted configuration flag.

To configure a directory server as untrusted, use the `dsreplication set-trust` command, as follows:

```
$ dsreplication --adminUID admin --adminPasswordFile pwd.txt -X \
  set-trust --trustedHost host1 --trustedPort 4444 \
  --modifiedHost host2 --modifiedPort 5444 --trustValue untrusted
```

The `dsreplication set-trust` command is supported in both interactive and non-interactive modes.

The configuration of trusted and untrusted servers is subject to the following restrictions:

- You can only configure the trust flag of a directory server from another trusted server in the topology. You cannot configure the trust flag from that server itself. The `-trustedHost` and `--modifiedHost` options can therefore not refer to the same directory server.
- When you modify a directory server from `untrusted` to `trusted`, the host that is being modified must be running, otherwise the command will fail.
- When you modify a directory server from `untrusted` to `trusted`, the host that is being modified must not contain any *untrusted changes*. An untrusted change is a change that has been made on an untrusted directory server and has therefore not been propagated to the rest of the topology. If the host that is being modified contains untrusted changes, the affected suffixes should be re-initialized with an appropriate data set from one of the trusted servers in the topology before the host is modified to trusted.
- If you modify the schema on an untrusted server, that server cannot be reconfigured as a trusted server. In this case, the server instance must be deleted and recreated.

Use the `dsreplication status` command to determine whether a directory server is trusted or untrusted. For example:

```
$ dsreplication status --adminUID admin --adminPasswordFile pwd.txt -X \
  --hostname host1 --port 4444
```

32.15.2 Understanding the Deployment Scenarios for Isolated Replicas

You can use isolated replicas in a replication topology to provide additional security in a demilitarized zone (DMZ) or to test client applications in a staging area.

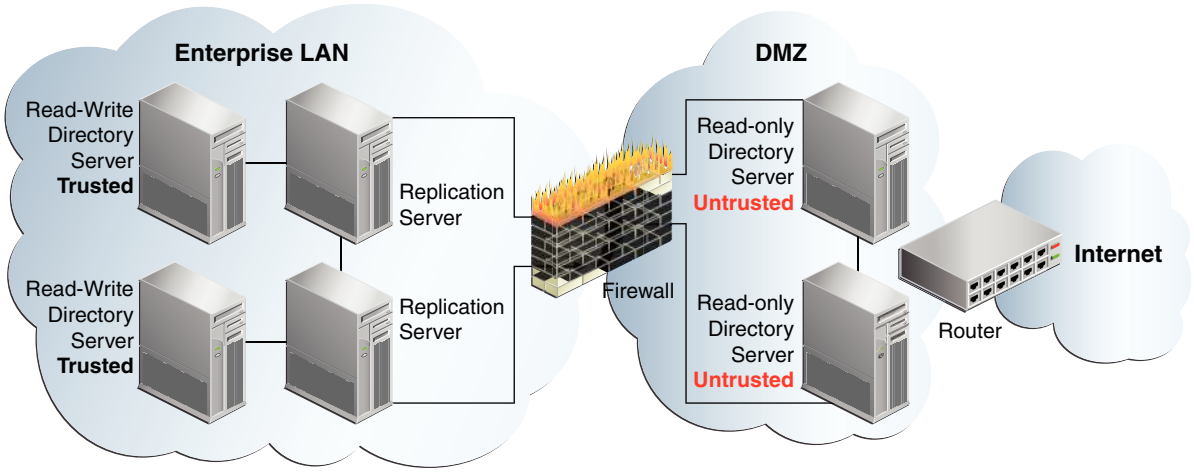
Note the following two scenarios for using isolated replicas in a replication topology:

- About Isolated Replicas in a DMZ
- About Isolated Replicas for Testing

32.15.2.1 About Isolated Replicas in a DMZ

A demilitarized zone (DMZ) is the area in an enterprise network that is exposed to an untrusted network, such as the Internet. A DMZ provides a layer of protection because it stands between a trusted and untrusted network. Direct access from the outside is limited to the equipment located inside the DMZ. The following figure shows how isolated replicas can be used in a DMZ.

Figure 32-3 Isolated Replicas in a Demilitarized Zone



By placing read-only directory servers in the DMZ, you can prevent compromised data from being transmitted to the replication servers in the private area of your network. When you deploy a replica in a DMZ, the replica is not protected by the enterprise firewall and might therefore at risk of being compromised. In such case, an unauthorized user might obtain access to the configuration of the replica and change it into a writable replica. Such a replica is therefore tagged as *untrusted* by the replication servers that are protected by the firewall.

Configuring the servers in the DMZ as untrusted safeguards against malicious data being accepted from them. The servers inside the private area are configured to have read and write access. This configuration ensures that data changes are propagated throughout the replication topology, only by the directory servers in the private area. The read-only directory servers in the DMZ obtain data changes from the replication servers located inside the private network. If an outside attacker attempts to compromise data, the direct access point is a read-only server inside the DMZ. Malicious data cannot be transmitted because directory servers in the DMZ are untrusted. The integrity of the server data inside the private enterprise LAN is therefore protected.

This scenario has the following configuration requirements:

- Each directory server in the DMZ is configured as untrusted *and* as read-only.
- Each replication server in the topology is located inside the private enterprise LAN.
- Each directory server in the private enterprise LAN is configured as a trusted server with read/write access.

Each trusted directory server in this topology has the following access rights:

- Can send changes to the replication server to which it is connected. Those changes will be propagated to all other directory servers in the topology.
- Can replay changes sent by the replication server to which it is connected.
- Can be the source of an online full update operation to initialize other servers with its data.

Each untrusted directory server in this topology has the following access limitations:

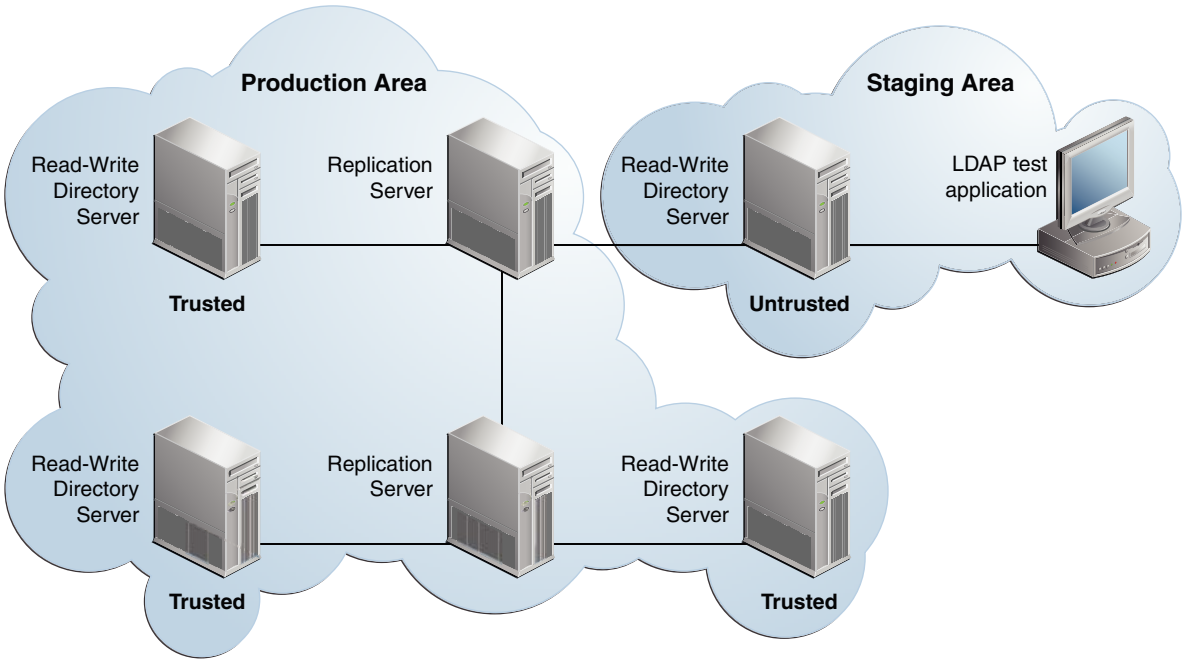
- Is not authorized to send changes to the replication server to which it is connected. If an untrusted directory server sends changes, the changes are evaluated as compromised data, and the replication server discards the changes.
- Can replay changes sent by the replication server to which it is connected.
- Cannot be the source of an online full update operation to initialize other servers with its data.

32.15.2.2 About Isolated Replicas for Testing

Isolated replicas can be useful to test an application against live data in a staging area. This can be accomplished by configuring the isolated replicas to be untrusted, but with read and write access. The application's access point is the isolated replica and data is written only to the isolated replicas in the staging area.

The following figure shows how isolated replicas can be used in a staging area.

Figure 32-4 Isolated Replicas in a Staging Area



32.16 Replicating Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory

Review these topics for the contextual information and instructions that are required for replicating the Oracle Directory Server Enterprise Edition data in the Oracle Unified Directory server.

- About Replicating Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory.
- Migrating the Oracle Directory Server Enterprise Edition Schema and Configuration.
- Configuring Replication Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory.
- Initializing the Oracle Unified Directory with Oracle Directory Server Enterprise Edition Data.

32.16.1 About Replicating Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory

Oracle Unified Directory provides a mechanism to replicate data between Oracle Directory Server Enterprise Edition and Oracle Unified Directory. The main purpose of this replication gateway is to enable migration from Oracle Directory Server Enterprise Edition to Oracle Unified Directory.

Setting up replication between these two disparate topologies involves three steps:

- Migrating the Oracle Directory Server Enterprise Edition schema and configuration to the Oracle Unified Directory server.
- Configuring replication between the Oracle Directory Server Enterprise Edition server and the Oracle Unified Directoryserver.
- Initializing the Oracle Unified Directory server with the data from the Oracle Directory Server Enterprise Edition server.

The following procedures describe each step. These procedures assume that you have the following:

- An installed and running Oracle Directory Server Enterprise Edition server.

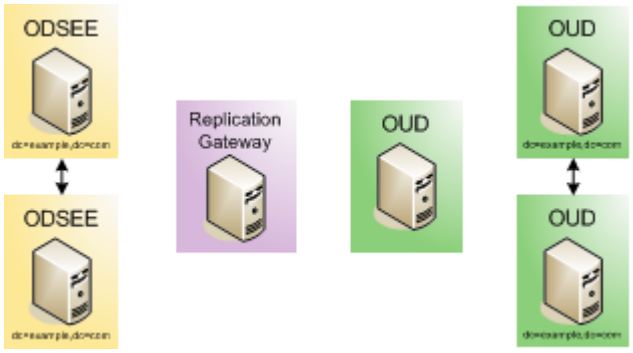
  The Oracle Unified Directory replication gateway supports the DS6-mode password policy only. If your Oracle Directory Server Enterprise Edition instance is using a DS5-mode password policy, you must update it.

- An installed and running Oracle Unified Directory directory server.

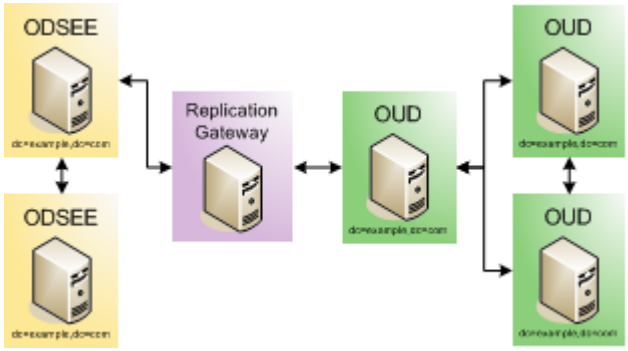
  The Oracle Unified Directory server must be configured *without* any suffixes, because that server is initialized with the data from the Oracle Directory Server Enterprise Edition server.

  If you have an existing, replicated Oracle Unified Directory topology, create an additional Oracle Unified Directory server instance, with no suffixes, and attach that server to the replication gateway. All `ds2oud` commands should be run on that empty Oracle Unified Directory server. When replication is working between the Oracle Directory Server Enterprise Edition server and the Oracle Unified Directory server, you can add the Oracle Unified Directory server to the existing replicated Oracle Unified Directory topology.

For example, assuming an existing Oracle Unified Directory topology, your server layout prior to migration would be as follows:



After migration, your server layout would be as follows:



32.16.2 Migrating the Oracle Directory Server Enterprise Edition Schema and Configuration

Oracle Unified Directory allows migration of the configuration and the schema of Sun ONE Directory Server 5.2, Sun Java System Directory Server Enterprise Edition 6.3.1, Sun Directory Server Enterprise Edition 7.0, and Oracle Directory Server Enterprise Edition 11g Release 1 (11.1.1) including all patchsets. The migration of this types of instances can be done using the `ds2oud` command tool.

The support of these versions of directory is only available for the tool `ds2oud`, but it does not apply to the use of the replication gateway which still requires at least an Oracle Directory Server Enterprise Edition 11g Release 1 (11.1.1).

In other words, depending on the instance version you migrate, the resulting Oracle Unified Directory instance requires supplementary manual steps to be fully functional, including modifying the data with respect to objectclasses and password policies, and converting metadata. However, if you run at least Oracle Directory Server Enterprise Edition 11g Release 1 (11.1.1), then it automatically takes care of data conversions while exporting the user data as described in Step 2.a in this section.

The procedure in this section describes various options of the `ds2oud` command. You can run the `ds2oud` command completely interactively by typing `ds2oud` on the command line. In interactive mode, the command prompts you for the required responses. For more information, see ds2oud.

1. On the Oracle Directory Server Enterprise Edition directory server, run the `ds2oud --diagnose` command, providing the connection details of the Oracle Directory Server Enterprise Edition server. The `ds2oud` command is located in `instance_dir/OUD/bin` for Linux and `instance_dir\OUD\bat` for Windows.

   This command assesses the Oracle Directory Server Enterprise Edition server instance and informs you whether any of the server configuration must be migrated to the Oracle Unified Directoryserver.

   ```
   $ ds2oud --diagnose -h host1.example.com -p 1389 \
     -D "cn=directory manager" -j pwdfile
   ```

   The `--diagnose` subcommand identifies the following elements of an Oracle Directory Server Enterprise Edition configuration:

   - any enabled user plug-ins
   - enabled subtree entry counter plug-ins (subtree entry counter plug-ins are not supported in Oracle Unified Directory)
   - extensions to the default schema
   - any CoS or role definitions
   - macro ACIs
   - ACI syntax validity
   - the type of password policy (only DS6-mode is supported)
   - conflicting entries in the data
   - encrypted attributes (attribute encryption is not supported in Oracle Unified Directory)

2. To verify data compliance regarding the Oracle Unified Directory schema:

   a. Export the Oracle Directory Server Enterprise Edition data to LDIF.

      On the Oracle Directory Server Enterprise Editionserver, run the `dsconf export` command as shown in the following example:

      ```
      $ dsconf export -f opends-export -h host1.example.com -p 1389 \
        dc=example,dc=com odsee-data.ldif
      ```

      > **Note:**
      >
      > The option `-f opends-export` in the preceding command is only applicable for Oracle Directory Server Enterprise Edition 11*g* Release 1 (11.1.1).

   b. When you have exported the data to LDIF, run the `ds2oud` command on the Oracle Unified Directory. For example:

      ```
      $ ds2oud --ldifDBFile odsee-data.ldif --userSchemaFile 99-user.ldif
      ```

      where `odsee-data.ldif` is the Oracle Directory Server Enterprise Editiondata exported to LDIF and `99-user.ldif` is the customized Oracle Directory Server Enterprise Editionschema file, if you have customised the Oracle Directory Server Enterprise Edition schema.

      This command highlights any schema inconsistencies between the Oracle Directory Server Enterprise Editiondata and the Oracle Unified Directory schema. Any schema extensions required by the Oracle Directory Server Enterprise Edition data must be added to the Oracle Unified Directory schema before you migrate the data.

3. Run the `ds2oud` command with one or more of the migration options to migrate the schema, the server configuration, or both.

   You must migrate the schema *before* you migrate the configuration, so that Oracle Unified Directory can validate the data.

   a. Running `ds2oud --migrateUserSchema` adds the Oracle Directory Server Enterprise Edition user schema (usually located in the file `99-user.ldif`) to the Oracle Unified Directory schema.

      If you plan to replicate collective attributes or password policy features in Oracle Unified Directory, you must prepare the Oracle Directory Server Enterprise Editionschema for these features.Oracle Unified Directory provides a customer schema file (`99OudSchemaExtract.ldif`) located in: `install-dir/OracleUnifiedDirectory/config/ds2oud` that enables you to add Oracle Unified Directory-specific schema elements to an Oracle Directory Server Enterprise Editioninstance.

      For information about adding this schema file to the Oracle Directory Server Enterprise Edition schema, see "*Extending Schema With a Custom Schema File*" in *Oracle Directory Server Enterprise Edition Administration Guide*.

   b. Running `ds2oud --migrateConfiguration` does the following:

      - Creates the naming contexts based on the existing Oracle Directory Server Enterprise Edition suffixes. You can specify whether the naming contexts are created in a single shared workflow element (`userRoot`) or in a workflow element per suffix. If the configuration includes sub-suffixes, one workflow element per suffix is imposed.
      - Migrates certain global configuration parameters that apply to Oracle Unified Directory, including `size-limit`, `lookthrough-limit`, `idle-time-limit`, `max-psearches`, and `bind-with-dn-requires-password`.
      - Migrates the global and back-end `allidsthreshold` parameters to the Oracle Unified Directory`index-entry-limit` back-end property.
      - Adds any configured indexes, and migrates specific `allidsthreshold` parameters on the index or index type to the new indexes.
      - Translates the DSE ACI into `ds-cfg-global-aci`, and checks the validity of ACIs by using Oracle Unified Directorysyntax validation.
      - Migrates the plug-in configuration if possible for the following plug-ins: 7-bit check, UID uniqueness, Referential Integrity, Strong password policy check.
      - Sets up a password policy and configures the default password policy to be equivalent to the default Oracle Directory Server Enterprise Editionpassword policy.

      > **Note:**
      >
      > Migration is possible only for Oracle Directory Server Enterprise Edition servers that are using a DS6-mode password policy.

   c. To migrate the schema and the configuration parameters, run the following command:

      ```
      $ ds2oud --migrateAll -D "cn=directory manager" -j pwdfile \
        -h host1.example.com -p 1389 \
      ```

```
--oudBindDN "cn=directory manager" --oudBindPasswordFile pwdfile \
--oudHostname localhost --oudAdminPort 4444 --oudPort 1389
```

where `-D`, `-j -h` and `-p` specify the connection parameters of the Oracle Directory Server Enterprise Editioninstance.

Most ACIs are stored in the entries themselves, and are therefore migrated when you export the data from the Oracle Directory Server Enterprise Edition instance and import it to the Oracle Unified Directory instance. The `--migrateAll` subcommand migrates only global ACIs that are stored in the configuration.

You are prompted for additional information relating to the Oracle Unified Directory configuration. This command creates a compatible configuration on the Oracle Unified Directory directory server.

32.16.3 Configuring Replication Between Oracle Directory Server Enterprise Edition and Oracle Unified Directory

You must install and configure the replication gateway between Oracle Directory Server Enterprise Edition and Oracle Unified Directory.

Install and configure the replication gateway, as described in "Setting Up the Replication Gateway" section in *Installing Oracle Unified Directory*.

At this point you must configure a global administrator on the Oracle Unified Directory server, for replication. If you intend to connect this server to an existing replicated Oracle Unified Directory topology at a later stage, use the same global administrator credentials that you have defined on the other Oracle Unified Directoryservers.

32.16.4 Initializing the Oracle Unified Directory with Oracle Directory Server Enterprise Edition Data

You can initialize the replicated data from Oracle Directory Server Enterprise Edition on the Oracle Unified Directory server and test the replication.

To initialize the Oracle Unified Directory with Oracle Directory Server Enterprise Edition data:

1. Prepare the Oracle Unified Directory server to be initialized. For example:

   ```
   $ dsreplication pre-external-initialization -h localhost -p 4444 \
     --adminUID admin --adminPasswordFile pwd.txt --baseDN dc=example,dc=com \
     -X -n --noPropertiesFile
   ```

2. On the Oracle Directory Server Enterprise Edition server, run the following command to export the data set:

   ```
   $ dsadm export -f opends-export dsee-instance-path baseDN exportedLDIFPath
   ```

   where *exportedLDIFPath* is the path of the resulting LDIF file that contains the replicated data.

   If the Oracle Directory Server Enterprise Edition data includes encrypted attributes, decrypt them with the `--decrypt-attr` option.

> **Note:**
>
> `dsadm export` creates a file in LDIF format.
>
> `dsadm backup` creates a binary copy of the database files of the Oracle Directory Server Enterprise Edition server. Because the database implementations of Oracle Directory Server Enterprise Edition and Oracle Unified Directory are very different, you cannot use the binary copy to export data from one server type to another.

3. Copy the LDIF file that was generated in step 2 to a directory that is accessible by the Oracle Unified Directory server. Ensure that the file permissions on the LDIF file allow read access by the server.
4. On the Oracle Unified Directory server, import the LDIF data, as follows:

```
$ import-ldif -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
  --includeBranch dc=example,dc=com --ldifFile path/to/exportedLDIFFile \
  --clearBackend --trustAll --noPropertiesFile
```

> **Note:**
>
> If you use a relative path to the LDIF file, the root for the relative path is the instance root, rather than the current working directory. So, for example, a path of `imports/odsee-data.ldif` here refers to `instance-root/imports/odsee-data.ldif`.

When you use the `opends-export` option during migration, DSEE-specific attributes might exist in some entries, preventing these entries from being imported. For instance, `nds5replconflict` might exist in the Oracle Directory Server Enterprise Edition data.Therefore, it is imperative to filter this attribute during import to Oracle Unified Directory using the following import option:

```
--excludeAttribute "nsds5replconflict"
```

5. Run the post-initialization script on the Oracle Unified Directory server, for example:

```
$ dsreplication post-external-initialization -h localhost -p 4444 \
  --adminUID admin --adminPasswordFile pwd.txt --baseDN dc=example,dc=com \
  -X -n --noPropertiesFile
```

6. To test that replication is working correctly, modify at least one entry on each Oracle Directory Server Enterprise Edition server and check the modification on the Oracle Unified Directory server.

33
Managing Directory Schema

Oracle Unified Directory provides a directory schema that includes several object classes and attributes. You can extend the schema provided with the directory server by creating new object classes and attributes.
Topics:

- Understanding Schema in Oracle Unified Directory
- Configuring Schema Checking
- Working With Object Identifiers (OIDs)
- Extending the Schema
- About Replicating the Schema
- Managing the Schema Using OUDSM

For detailed information about specific schema elements, see Understanding the Oracle Unified Directory Schema Model.

33.1 Understanding Schema in Oracle Unified Directory

The *schema* defines and governs the types of information objects that can be stored in a directory. A schema defines the types of entries in the directory information tree, maintains element uniqueness, and prevents unchecked schema growth that can arise when new elements are added to the directory.

This section covers the following topics:

- About Oracle Unified Directory Schema
- Designing and Extending the Schema
- Default Schema Files

33.1.1 About Oracle Unified Directory Schema

A directory server instance reads the schema once at startup and then uses the schema information to match a search filter request or assertion to an entry's attributes to determine if any add or modify operations are permitted by the client.

In most cases, the default schema should be sufficient for most applications. However, you can take advantage of the flexibility of the directory server to extend the schema to suit your applications. The general procedure is not to relinquish the standard schema to a new custom schema, but to use the standard attributes or object classes wherever possible. If you require custom attributes or object classes that are not handled with the standard schema, you can create or extend the standard schema with auxiliary attributes and object classes required for your application.

The schema is stored in the directory under the suffix (`cn=schema`). The directory server also has a subschema subentry that defines the schema elements plus the set of operational attributes in the directory.

You can extend the schema in one of two ways:

- Extend the schema over LDAP.
- Create a custom schema definition file.

33.1.2 Designing and Extending the Schema

Before you consider extending the default schema, or designing your own schema, ensure that you have a solid understanding of schema syntax and design.

The basic steps to design or extend a schema are as follows:

1. Map the data to the default schema. Where possible, use the existing schema elements that are defined in the directory server. Standard schema elements help to ensure compatibility with directory-enabled applications. Because the schema is based on the LDAP standard, it has been reviewed and agreed upon by a large number of directory users.
2. Identify unmatched data. The default schema was designed to accommodate a large variety of information objects. However, if the schema does not handle your specific data type, then make note of it and any other data types needed for your directory.
3. Extend the default schema to define new elements. For optimal performance, reuse existing schema elements wherever possible. Also, minimize the number of mandatory attributes that you define for each object class. Keep the schema as simple as possible. Do not define more than one object class or attribute for the same purpose.
4. Use schema checking. Schema checking ensures that attributes and object classes conform to the schema rules.
5. Select and apply a consistent data format. The LDAP schema allows you to place any data on any attribute value. However, you should store data consistently by selecting a format appropriate for your LDAP client application and directory users.

33.1.3 Default Schema Files

The default schema provided with the directory server is a collection of LDIF files stored in `OUD_ORACLE_HOME/config/schema`. These schema files are applied to every server instance that is associated with that `OUD_ORACLE_HOME`.

A directory server instance loads the schema files in alphanumeric order (numerals first) at server startup.

> **Caution:**
>
> Never modify the standard schema definitions and internal operational attributes in these files.

The following table describes the default schema files and their contents.

Table 33-1 Default Schema Files

Schema File	Description
00-core.ldif	Contains the schema definitions for the LDAPv3 standard user and organization.
01-pwpolicy.ldif	Contains the schema definitions for password policies based on the draftldappolicy draft.
02-config.ldif	Contains the schema definitions for the attribute and object class definitions in the directory configuration file.
03-changelog.ldif	Contains the schema definitions for storing changes to directory data based on the draftldap-changelog.
03-rfc2713.ldif	Contains the schema definitions for representing Java objects in an LDAP directory based on RFC 2713.
03-rfc2714.ldif	Contains the schema definitions for representing CORBA object references in an LDAP directory based on RFC 2714. The Common Object Request Broker Architecture (CORBA) integrates machines in a multivendor, multiplatform environments using CORBA objects. A directory server can be a repository for CORBA object references, which allow for a centrally administered service for CORBA-compliant applications.
03-rfc2739.ldif	Contains the schema definitions for representing calendar attributes for a vCard directory based on RFC 2739. Calendar applications require a calendar user agent to locate a URI, located in a directory, for an individual's calendar. **Note**: The definition in RFC 2739 contains some errors. This schema file has been altered from the standard definition to fix some those problems.
03-rfc2926.ldif	Contains the schema definitions for mapping Service Location Protocol (SLP) advertisements based on RFC 2926. This specification allows directory servers to serve SLP directory agent back ends that create mappings between SLP templates and the LDAP directory schema.
03-rfc3112.ldif	Contains the schema definitions for the authentication password syntax based on RFC 3112.
03-rfc3712.ldif	Contains the schema definitions for storing printer information in the directory based on RFC 3712.
03-uddiv3.ldif	Contains the schema definitions for storing UDDI v3 information in the directory based on RFC 4403. Universal Description, Discovery and Integration (UDDI) is a platform-independent, XML-based registry for companies on the Internet. UDDI enables companies to publish service listings and defines which software applications interact together over the Internet.
04-rfc2307bis.ldif	Contains the schema definitions for storing naming service information in the directory based on draftrfc2307bis.
05-rfc4876.ldif	Contains schema definitions from RFC 4876, which defines a schema for storing Directory User Agent (DUA) profiles and preferences.
05-solaris.ldif	Contains schema definitions required for Solaris and OpenSolaris LDAP naming services.
06-compat.ldif	Contains the attribute type and objectclass definitions for use with the directory server configuration.
10-ad-paging.ldif	Contains schema definitions required for the Active Directory paging function.
10-distribution.ldif	Contains the schema definitions required for the distribution functionality of a proxy server instance.
10-global-index-catalog.ldif	Contains the schema definitions required for the global indexing functionality of a proxy server instance.

Table 33-1 (Cont.) Default Schema Files

Schema File	Description
10-loadbalancing.ldif	Contains the schema definitions required for the load balancing functionality of a proxy server instance.
10-proxy.ldif	Contains the schema definitions specific to a proxy server instance.
10-replication-gateway.ldif	Contains the schema definitions specific to a replication gateway server instance.
10-virtualization.ldif	Contains the schema definitions required for the virtualization functionality of a proxy server instance.

33.2 Configuring Schema Checking

Oracle Unified Directory provides a schema-checking mechanism that verifies whether newly-written or added entries conform to the directory server's schema. This mechanism ensures that data imported using `import-ldif`, or added using `ldapmodify`, meets the syntax rules of the schema.

The schema checking configuration is part of the advanced global configuration, and can be displayed with the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  --advanced get-global-configuration-prop
Property                                : Value(s)
----------------------------------------:----------------------------------------
...
check-schema                            : true
...
invalid-attribute-syntax-behavior       : reject
...
single-structural-objectclass-behavior : reject
...
```

The following configuration properties control schema-checking:

- `check-schema`. Possible values: `true` (default), `false`. This property controls whether the directory server should do schema-checking on newly imported or added entries. By default, the property is set to true. If you must tune the server for maximum performance and you are certain that your clients will never make a change that causes a schema violation, then you can set the property to `false`. The small performance benefits are minimal compared to the potential risks to your directory.
- `invalid-attribute-syntax-behavior`. Possible values are: `reject` (default), `accept`, and `warn`. This property controls how the server should behave if an attempt is made to use an attribute value that violates the associated syntax. By default, the server rejects any requests to use attributes that violate the schema. If this property is set to `accept`, then the server silently accepts attribute violations. If this attribute is set to `warn`, the server accepts violations, but writes a message to the error log. If the `check-schema` property is set to `false`, invalid attribute syntax checking is not enforced.
- `single-structural-objectclass-behavior`. Possible values are: `reject` (default), `accept`, and `warn`. This property controls how the server should behave if an attempt is made to create or alter an entry that does not have exactly one structural object class. This means that object classes with no structural object classes or more than one are rejected

by default. If this property is set to `accept`, entries with no structural object classes are allowed. If this property is set to `warn`, entries with no structural object classes (or more than one) are allowed, but a message is written to the error log. If the `check-schema` property is set to `false`, single structural object class checking is not enforced.

> **Caution:**
>
> Changing the value of these properties from the default puts the integrity of the schema at risk, so you should generally *not* alter these values.

33.3 Working With Object Identifiers (OIDs)

An object identifier (OID) is a numeric string used to uniquely identify an object in a directory. OIDs are used in directory schema, controls, and extended operations that require unique identification of elements.

This section covers the following topics:

- About Object Identifiers (OIDs).
- Obtaining a Base OID.

33.3.1 About Object Identifiers (OIDs)

LDAP object classes and attributes require a base object identifier (OID) that must be unique within your organization to avoid naming conflicts in the directory.

If you plan to use your directory internally within your organization, use the OIDs provided in the directory server. If you plan to export your schema or publicly expose your schema in any way, consider entering a request for a unique OID for your organization. For more information, see Obtaining a Base OID.

After you have obtained a base OID, you can add branches to it for your organization's object classes and attributes. For example, the directory server uses an assigned base OID of `1.3.6.1.4.1.26027`. For each component type, the directory server provides unique branch numbers to the base OID for each schema component.

Oracle Unified Directory provides a comprehensive set of OIDs that should be sufficient for most applications.

The following table shows the base OIDs used for each schema component:

Table 33-2 Base OIDs Used for Each Schema Component

OID Value	Type
`1.3.6.1.4.1.26027.1.1`	Attribute
`1.3.6.1.4.1.26027.1.2`	Object classes
`1.3.6.1.4.1.26027.1.3`	Attribute syntaxes
`1.3.6.1.4.1.26027.1.4`	Matching rules
`1.3.6.1.4.1.26027.1.5`	Controls
`1.3.6.1.4.1.26027.1.6`	Extended operations

Table 33-2 (Cont.) Base OIDs Used for Each Schema Component

OID Value	Type
1.3.6.1.4.1.26027.1.9	General use
1.3.6.1.4.1.26027.1.999	Experimental use

For each schema type, a unique branch number is added to the base OID. For example, attribute types use a branch number of `1` to form the OID of `1.3.5.1.4.1.26027.1.*1*`. For each specific attribute type, the directory server assigns another set of branch numbers, one for each attribute type.

The following table displays a (partial) list of assigned OID values for attribute types.

Table 33-3 Assigned OID Values for Attribute Types

OID Value	Attribute Type
1.3.6.1.4.1.26027.1.1.1	ds-cfg-java-class
1.3.6.1.4.1.26027.1.1.2	ds-cfg-enabled
1.3.6.1.4.1.26027.1.1.3	ds-cfg-allow-attribute-name-exceptions
1.3.6.1.4.1.26027.1.1.4	ds-cfg-allowed-client
1.3.6.1.4.1.26027.1.1.5	ds-cfg-allow-ldap-v2

Oracle Unified Directory allows the use of non-numeric OIDs if a corresponding numeric OID is defined within the schema. For example, you can use a non-numeric OID, `mytestattribute-oid` for the named attribute, `myTestAttribute`. The non-numeric OID must be all lowercase with the `-oid` appended to the named attribute. The use of non-numeric OIDs is an LDAP-specification violation but is permissible for ease of use.

33.3.2 Obtaining a Base OID

If you plan to make your directory server publicly available, or if you plan to redistribute your schema definitions for custom applications, you can obtain a base OID for your organization. You can use your own OIDs in a custom schema file if you plan to create custom extensions to the directory server. Alternatively, you can modify the schema configuration files by adding your base OID with its respective branch number.

> **Note:**
>
> Do not modify the default OIDs unless you are sure of what you are doing. Modifying the OIDs can potentially damage your directory server.

To obtain and create base OIDs for your organization:

1. Point your browser to the Internet Assigned Numbers Authority (IANA) website at (`http://www.iana.org`) or a national organization in your country that handles such tasks. In some countries, corporations already have OIDs assigned to them. If your organization does not already have an OID, you can fill out a request at the IANA website.

2. Determine the unique object classes, attributes, names, and other schema elements. Ensure that the names are descriptive to make it easier to manage the schema. One trick is to add a custom prefix to your custom object classes and attributes. For example, if your organization is Example.com, you can add the prefix `Example` before each custom schema element, such as adding `Example` to a `Person` object class as in `ExamplePerson`.
3. Create an OID registry to keep track of OID assignments. The registry is nothing more than a list that you maintain to ensure that OIDs and their descriptions are unique within your directory. The registry should be sufficiently protected so that only a privileged administrator can modify the registry.
4. Create branches in the OID tree to accommodate the schema elements.
5. Shut down the directory servers in your topology.
6. Manually edit the schema configuration files on each directory server in your topology. Replace each OID with your company's OID. This avoids problems with schema replication seeing differences in the schema and attempting to synchronize the information.
7. Manually edit any custom schema extensions. Ideally, you should define any custom extensions in a separate file.

33.4 Extending the Schema

Oracle Unified Directory supports multiple methods to extend the schema.

This section covers the following topics:

- About Extending the Schema
- Managing Attribute Types
- Managing Object Classes

33.4.1 About Extending the Schema

Oracle Unified Directory supports multiple methods to extend the schema. The standard schema files are a set of LDIF files located in `OUD_ORACLE_HOME/config/schema`. Do not modify these files directly, because doing so can result in unpredictable server behavior.

The standard schema definitions apply to every server instance associated with that `OUD_ORACLE_HOME`. Custom schema definitions located in *`instance-dir`*`/OUD/config/schema/99-user.ldif` apply only to the server instance in which they are created.

You can extend the schema as follows:

- **Extend the schema over LDAP.** Define your schema extensions, write the definitions to an LDIF file, and add the custom schema extensions by using the `ldapmodify` command.

  When you use this method, the directory server automatically writes the new schema definitions to the file:

  `instance-dir``/OUD/config/schema/99-user.ldif`

  To specify a different schema file, include the `X-SCHEMA-FILE` element with the name of your schema file. For example, as part of your attribute type definition, include the element `X-SCHEMA-FILE '98myschema.ldif'`.

  When you extend the schema over LDAP, you do not need to restart the server to take the schema modifications into account.

- **Create a custom schema file.** Create a custom schema file with your definitions and move the file to the directory:

  instance-dir/OUD/config/schema/

  The directory server loads schema files in alphanumeric order with numbers loaded first. As such, you should name custom schema files as follows: [00-99]filename.ldif. The number should be higher than any standard schema file that has already been defined. If you name custom schema files with a number that is lower than the standard schema files, the server might encounter errors when loading the schema.

  When you extend the schema with a custom schema file, the server must be restarted before the schema modifications are taken into account.

- **Modify an existing schema file.** You can add a custom schema extension to an existing custom schema file, such as *instance-dir*/OUD/config/schema/99-user.ldif.

  When you extend the schema by modifying an existing schema file, the server must be restarted before the schema modifications are taken into account.

When you add new schema elements, all attributes must be defined before they can be used in an object class. If you are creating several object classes that inherit from other object classes, you must create the parent object class first.

Each custom attribute or object class that you create should be defined in only one schema file.

When you define new schema definitions manually, the best practice is to add these definitions to the 99-user.ldif file or to your designated schema file.

33.4.2 Managing Attribute Types

You can add new attribute types to the schema by using the ldapmodify command. The attribute types syntax requires that you provide at least a valid OID to define a new element.

This section covers the following topics:

- List of Identifiers for Attribute Types.
- Viewing Attribute Types.
- Creating an Attribute Type.
- Deleting an Attribute Type.

33.4.2.1 List of Identifiers for Attribute Types

In typical applications, you can optionally include the following identifiers for the attribute type. To see the full set of attribute type elements, see Understanding Attribute Types.

OID
Required. Specifies the OID that uniquely identifies the attribute type in the directory server. The LDAP v3 specification requires the OID to be a UTF-8 encoded dotted decimal. However, Oracle Unified Directory supports the use of non-numeric OIDs for easy identification if the schema is used internally within the organization. The format is attributename-oid, for example, telephoneNumber-oid. Each non-numeric OID must have its corresponding dotted decimal OID defined in the schema.

NAME
Optional. Specifies the set of human-readable names that are used to refer to the attribute type. If there is a single name, enclose it in single quotes, for example, `'blogURL'`. If there are multiple names, enclose each name in single quotes separated by spaces, and then enclose the entire set of names within parentheses, for example, `('blog' 'blogURL')`. Ensure that there is a space between the left parenthesis and the name, and a space before the closing parenthesis.

SUP
Optional. Specifies the superior attribute type when you want one attribute type to inherit elements from another attribute type. The matching rule and attribute syntax specifications from the superior attribute type can be inherited by the subordinate type if it does not override the superior attribute type definition. The OID, any of the human-readable names associated with the superior attribute type or both can be used to collectively reference all of the subordinate attribute types.

DESC
Optional. Specifies a human-readable description of the attribute type.

SYNTAX
Optional. Specifies the attribute syntax for use with the attribute type. If provided, it should be given as a numeric OID. The core syntaxes are defined in section 3.3. of RFC 4517 (http://www.ietf.org/rfc/rfc4517.txt) and in Appendix A of the same document.

SINGLE-VALUE
Optional. Specifies whether the attributes of that type are allowed to have only a single value in any entry in which they appear. If `SINGLE-VALUE` is not present, the attributes are allowed to have multiple distinct values in the same entry.

NO-USER-MODIFICATION
Optional. Indicates that the values of the attributes of the given type cannot be modified by external clients (that is, the values can be modified only by internal processing within the directory server).

USAGE
Optional. Indicates how the attribute is to be used. Possible values are as follows: `userApplications`. Used to store user data. `directoryOperation`. Used to store data required for internal processing within the directory server. `distributeOperation`. Used to store operational data that must be synchronized across directory servers in the topology. `dSAOperation`. Used to store operational data that is specific to a particular directory server and should not be synchronized across the topology.

extensions
Optional. Specifies the extensions available to the attribute type.Oracle Unified Directory provides the following extensions:

- `X-ORIGIN`. Provides information on where the attribute type is defined. The element is a nonstandard tool that you can use to locate the schema element, for example, the RFC number (`RFC4517`).
- `X-SCHEMA-FILE`. Indicates which schema file contains the attribute type definition. Used for internal purposes only and is not exposed to clients. You can use this extension to specify where the directory server should store your custom schema definitions.

- `X-APPROX`. Indicates which approximate matching rule should be used for the attribute type. If specified, the value should be the name of the OID of a registered approximate matching rule.

For example, you can specify the addition of a new attribute type, `blogURL`, in an LDIF file that will be added to the schema.

```
$ cat blogURL.ldif
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.32473.1.1.590
  NAME ( 'blog' 'blogURL' )
  DESC 'URL to a personal weblog'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE
  X-ORIGIN 'Oracle Unified Directory Server'
  USAGE userApplications )
```

> **Note:**
>
> Pay special attention to the spaces in an attribute type declaration. The LDAP specification requires that a space exist between the opening parenthesis and the OID, and the value of the `USAGE` element and the closing parenthesis. Further, the LDIF specification states that LDIF parsers should ignore exactly one space at the beginning of each line. Therefore, it is a good practice to add two (2) spaces at the beginning of the line that starts with an element keyword. For example, add two spaces before `NAME`, `DESC`, `SYNTAX`, `SINGLE-VALUE`, `X-ORIGIN`, and `USAGE` in the previous example.
>
> The OIDs used in this example are for illustration purposes only and should not be implemented in your directory.

33.4.2.2 Viewing Attribute Types

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You can view the schema definitions by using the `ldapsearch` command. Schema elements are represented as LDAP subentries, and searches on `cn=schema` must therefore include the LDAP Subentry search control.

1. Use the `ldapsearch` command with the LDAP Subentry search control, as follows:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=schema" -s base "(objectclass=*)" attributeTypes
dn: cn=schema
attributeTypes: ( 2.5.4.41 NAME 'name' EQUALITY caseIgnoreMatch SUBSTR
 caseIgnoreeSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768}
 X-ORIGIN 'RFC 4519' )
attributeTypes: ( 2.5.4.49 NAME 'distinguishedName' EQUALITY
 distinguishedNameMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 X-ORIGIN
 'RFC 4519' )
attributeTypes: ( 2.5.4.0 NAME 'objectClass' EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 X-ORIGIN 'RFC 4512' )
...(more output)...
```

2. To view a specific attribute type, use the `--dontWrap` option and then use the `grep` command (on UNIX systems) to search for the required attribute.

The following example searches for attribute types that contain the string `telexNumber`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b cn=schema -s base --dontWrap "(objectclass=*)" \
  attributeTypes | grep "telexNumber"
  attributeTypes: ( 2.5.4.21 NAME 'telexNumber' SYNTAX 1.3.6.1.4.1.1466.115.121.1.52
X-ORIGIN 'RFC 4519' )
  attributeTypes: ( 2.5.4.21.1 NAME 'c-TelexNumber' SUP telexNumber COLLECTIVE X-
ORIGIN 'RFC 3671' )
```

33.4.2.3 Creating an Attribute Type

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You can add custom schema definitions by using the `ldapmodify` command. This example adds an attribute named `blog`.

1. Using a text editor, create an LDIF file with your schema extensions.

```
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.32473.1.1.590
  NAME ( 'blog' 'blogURL' )
  DESC 'URL to a personal weblog'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE
  X-ORIGIN 'Oracle Unified Directory Server'
  USAGE userApplications )
```

2. Use `ldapmodify` to add the file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -a -f blogURL.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

3. Verify the addition by displaying it using `ldapsearch`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b "cn=schema" -s base --dontWrap "(objectclass=*)" \
  attributeTypes | grep 'blog'
attributeTypes: ( 1.3.6.1.4.1.32473.1.1.590 NAME ( 'blog' 'blogURL' )
DESC 'URL to a personal weblog' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE X-ORIGIN 'Oracle Unified Directory Server' USAGE userApplications )
```

> **Note:**
>
> Oracle Unified Directory automatically adds new attribute definitions to the file *instance-dir*`/OUD/config/schema/99-user.ldif`.

33.4.2.4 Deleting an Attribute Type

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You can delete custom schema definitions by using the `ldapmodify` command.Oracle Unified Directory does not allow deletions to standard schema definitions.

> **Caution:**
>
> Be careful when deleting attribute types, because doing so can harm your directory. Do not delete an attribute type unless absolutely necessary.

1. Create the delete request in an LDIF file.

```
dn: cn=schema
changetype: modify
delete: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.32473.1.1.590
  NAME ( 'blog' 'blogURL' )
  DESC 'URL to a personal weblog'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE
  X-ORIGIN 'Oracle Unified Directory Server'
  USAGE userApplications )
```

2. Use the `ldapmodify` command to process the delete request.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --defaultAdd --fileName "remove_blogURL.ldif"
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

33.4.3 Managing Object Classes

Object classes are named sets of attribute definitions that are used to control the types of data stored in entries. You can add new object classes to the schema by using the `ldapmodify` command. The object class syntax requires that you provide at least a valid OID to define your new element.

This section covers the following topics:

- List of Optional identifiers for Object Classes.
- Viewing Object Classes.
- Creating an Object Class.
- Deleting an Object Class.

33.4.3.1 List of Optional identifiers for Object Classes

In typical applications, you will also include the following optional identifiers for the object class type. For more information about the object class definition, see Understanding Schema in Oracle Unified Directory.

`OID`
Required. Specifies the OID that uniquely identifies the object class in the directory server. The LDAP v3 specification requires the OID to be a UTF-8 encoded dotted decimal. However, Oracle Unified Directory supports the use of non-numeric OIDs for easy identification because the schema is used internally within the organization. For example, the format is *objectClassName*`-oid`, such as `person-oid`.

NAME
Optional. Specifies the set of human-readable names that are used to refer to the object class. If there is a single name, enclose it in single quotes, for example, `'blogURL'`. If there are multiple names, enclose each name in single quotes separated by spaces, and then enclose the entire set of names within parentheses, for example, `('blog' 'blogURL')`. Ensure that there is a space between the left parenthesis and the name, and a space before the closing parenthesis.

DESC
Optional. Specifies a human-readable description of the object class. If specified, the description should be enclosed in single quotation marks.

SUP
Optional. Specifies the superior object class when you want it to inherit elements from another object class. The directory server allows only one superior object class, although the LDAP v3 specification allows for multiple superior object classes.

OBSOLETE
Optional. Indicates whether the object class is active or not. If an object class is marked as `OBSOLETE`, then it should not be referenced by any new elements created in the directory server.

SUP oids
Optional. The `SUP` keyword should be followed by the OID of the superior class.

KIND
Optional. Indicates the type of object class that is being defined. Allowed values are `ABSTRACT`, `AUXILIARY` and `STRUCTURAL`.

MUST oids
Optional. Specifies the set of attribute types that are required to be present (that is, have at least one value) in entries with that object class. If there is only a single required attribute, then the `MUST` keyword should be followed by the name or the OID of that attribute type. If there are multiple required attribute types, then separate them with dollar signs ($) and enclose the entire set of attribute types in parentheses. For example, `MUST (sn $cn)`.

MAY oids
Optional. Specifies the set of attribute types that are allowed but not required to be present in entries with that object class. If there is only a single required attribute, then the `MAY` keyword should be followed by the name or the OID of that attribute type. If multiple required attribute types are specified, then separate them by dollar signs (`$`) and enclose the entire set of attribute types in parentheses. For example, `MAY (userPassword $telephoneNumber $seeAlso $description)`.

extensions
Optional. Specifies the extensions available to the object class. The directory server provides the following extensions: `X-ORIGIN`. Provides information on where the object class is defined. The element is a nonstandard tool that the user can use to conveniently locate the schema element. `X-SCHEMA-FILE`. Indicates which schema file contains the object class definition. Used for internal purposes only and is not exposed to clients. You can use this extension to specify where the directory server is to store your custom schema definitions.

For example, you can specify the addition of a new object class, `blogger`, in an LDIF file to be added to the schema.

```
$ cat blogger.ldif
dn: cn=schema
```

```
changetype: modify
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.32473.1.1.10
  NAME ( 'blogger' )
  DESC 'Someone who has a blog'
  SUP inetOrgPerson
  STRUCTURAL
  MAY blog
  X-ORIGIN 'Oracle Unified Directory Server' )
```

Pay special attention to the spaces in your object class declaration. The LDAP specification requires that a space exist between the opening parenthesis and the OID, and the value of the `X-ORIGIN` element and the closing parenthesis. Further, the LDIF specification states that LDIF parsers should ignore exactly one space at the beginning of each line. Therefore, it is a good practice to add two spaces before the line that begins with an element keyword, such as, `NAME`, `DESC`, `SUP`, `STRUCTURAL`, `MAY`, and `X-ORIGIN` in the previous example.

The OIDs used in this example are for illustration purposes only and should not be implemented in your directory.

33.4.3.2 Viewing Object Classes

The `cn=schema` entry has a multivalued attribute, `objectClasses`, that contains definitions of each object class in the directory schema. You can view the schema definitions by using the `ldapsearch` command.

1. Use the `ldapsearch` command to view object class definitions.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b cn=schema -s base "(objectclass=*)" objectClasses
dn: cn=schema
objectClasses: ( 2.5.6.0 NAME 'top' ABSTRACT MUST objectClass X-ORIGIN
 'RFC 4512' )
objectClasses: ( 2.5.6.1 NAME 'alias' SUP top STRUCTURAL MUST aliasedObjectName
 X-ORIGIN 'RFC 4512' )
objectClasses: ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL MUST c MAY
 ( searchGuide $ description ) X-ORIGIN 'RFC 4519' )
objectClasses: ( 2.5.6.3 NAME 'locality' SUP top STRUCTURAL MAY ( street $
 seeAlso $ searchGuide $ st $ l $ description ) X-ORIGIN 'RFC 4519' )
objectClasses: ( 2.5.6.4 NAME 'organization' SUP top STRUCTURAL MUST o MAY
 ( userPassword $ searchGuide $ seeAlso $ businessCategory $ x121Address $
 registered Address $ destinationIndicator $ preferredDeliveryMethod $
 telexNumber $ teletexTerminalIdentifier $ telephoneNumber $
 internationaliSDNNumber $ facsimileTelephoneNumber $ street $ postOfficeBox $
 postalCode $ postalAddress $ physicalDeliveryOfficeName $ st $ l $ description
 ) X-ORIGIN 'RFC 4519' )
...(more output)...
```

2. Use the `--dontWrap` option and the `grep` command to search for a specific object class.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b cn=schema -s base --dontWrap "(objectclass=*)" \
  objectClasses | grep "inetOrgPerson"
objectClasses: ( 2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson' SUP
organizationalPerson
STRUCTURAL MAY ( audio $ businessCategory $ carLicense $ departmentNumber $
displayName
$ employeeNumber $ employeeType $ givenName $ homePhone $ homePostalAddress $
initials
$ jpegPhoto $ labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $ roomNumber
```

```
$ secretary $ uid $ userCertificate $ x500UniqueIdentifier $ preferredLanguage
$ userSMIMECertificate $ userPKCS12 ) X-ORIGIN 'RFC 2798' )
```

33.4.3.3 Creating an Object Class

The `cn=schema` entry has a multivalued attribute, `objectClasses`, that contains definitions of each object class in the directory schema. You add custom schema by using the `ldapmodify` command. This example adds an object class `blogger` based on the attribute type that was created in the previous example.

1. Using a text editor, create an LDIF file with your schema extensions.

```
dn: cn=schema
changetype: modify
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.32473.1.1.10
  NAME ( 'blogger' )
  DESC 'Someone who has a blog'
  SUP inetOrgPerson
  STRUCTURAL
  MAY blog
  X-ORIGIN 'Oracle Unified Directory Server' )
```

2. Use the `ldapmodify` command to add the file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -a -f blogger.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

3. Verify the addition by displaying it with `ldapsearch`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  -b cn=schema -s base --dontWrap "(objectclass=*)" \
  objectClasses | grep 'blogger'
```

> **Note:**
>
> Oracle Unified Directory automatically adds new object class definitions to the file *instance-dir*`/OUD/config/schema/99-user.ldif`.

33.4.3.4 Deleting an Object Class

The `cn=schema` entry has a multivalued attribute, `objectClasses`, that contains definitions for each object class in the directory schema. You can delete custom object class definitions by using the `ldapmodify` command.

> **Caution:**
>
> Be careful when deleting object classes, because doing so can harm your directory. Do not delete an object class unless absolutely necessary.

1. Create the delete request in LDIF format.

```
dn: cn=schema
changetype: modify
delete: objectClasses
objectClasses: ( 1.3.6.1.4.1.32473.1.1.10
  NAME ( 'blogger' )
  DESC 'Someone who has a blog'
  SUP inetOrgPerson
  STRUCTURAL
  MAY blog
  X-ORIGIN 'Oracle Unified Directory Server' )
```

2. Remove the object class by using `ldapmodify` to apply the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -j pwd-file \
  --fileName "remove_objectclass_schema.ldif"
```

33.5 About Replicating the Schema

In a replicated topology, schema definitions are automatically replicated to ensure that all servers use a single schema. Schema modifications on any server are replicated to all other servers in the topology.

When you configure replication, the schema of the first server is used to initialize the schema of the second server by default. You can, however, specify that the schema of the second server be used to initialize the schema of the first server. You can also specify that schema replication be disabled altogether. For more information, see Configuring Schema Replication.

33.6 Managing the Schema Using OUDSM

You can manage most elements of the directory schema with OUDSM. The following topics indicate the steps to manage the most common aspects of viewing and extending the schema.

This section covers the following topics:

- Adding a New Attribute Type.
- Adding an Attribute Based on an Existing Attribute.
- Modifying an Attribute.
- Deleting an Attribute.
- Viewing All Directory Attributes.
- Searching for Attributes.
- Viewing the Indexing Details of an Attribute.
- Adding a New Object Class.
- Adding an Object Class Based on an Existing Object Class.
- Viewing the Properties of an Object Class.
- Modifying an Object Class.
- Deleting an Object Class.
- Searching for Object Classes.
- Displaying a List of LDAP Syntaxes.
- Searching for a Syntax.
- Displaying a List of LDAP Matching Rules.

- Searching for a Matching Rule.
- Displaying a List of Content Rules.
- Searching for a Content Rule.
- Creating a New Content Rule.
- Creating a Content Rule Based on an Existing Content Rule.
- Modifying a Content Rule.
- Deleting a Content Rule.

33.6.1 Adding a New Attribute Type

Use Oracle Unified Directory Services Manager (OUDSM) to add a new attribute type to the schema.

To add a new attribute type to the schema by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. Click the **Add** icon.
5. Complete the following information on the **Create new attribute** window:
   - **Name.** Enter a unique name for the new attribute type.
   - **Object ID.** Specify the OID that uniquely identifies the attribute type in the directory server.Oracle Unified Directory supports the use of non-numeric OIDs for easy identification if the schema is used internally within the organization. However, for this release OUDSM supports numeric OIDs only.
   - **Description.** Enter a human-readable description of the attribute type.
   - **Syntax.** Enter the attribute syntax for use with the attribute type. If provided, the syntax should be specified as a numeric OID. The core syntaxes are defined in section 3.3. of RFC 4517 and in Appendix A of the same document.
   - **Size.** Enter a maximum size for the value of the attribute, in bytes. For multi-valued attributes, this setting refers to the maximum size of a single value, not of the combined values.
   - **Usage.** Specify how the attribute will be used. Possible values are as follows:
     - **userApplications.** The attribute will be used to store user data.
     - **directoryOperation.** The attribute will be used to store data that is required for internal processing within the directory server.
     - **distributedOperation.** The attribute will be used to store operational data that must be synchronized across directory servers in the topology.
     - **dSAOperation.** The attribute will be used to store operational data that is specific to a particular directory server and should not be synchronized across the topology.
   - **Ordering.** Select the ordering matching rules for this attribute type. See Understanding Matching Rules.

- **Equality.** Select the equality matching rules for this attribute type. See Understanding Matching Rules.
- **Substring.** Select the substring matching rules for this attribute type. See Understanding Matching Rules.
- **Obsolete.** Select this box if the attribute type is no longer in use but is retained for compatibility.
- **Single Value.** Indicate whether attributes of this type may have only a single value in any entry in which they appear. If this checkbox is not selected, the attributes may have multiple distinct values in the same entry.
- **Collective.** Indicate whether the attribute is a collective attribute. For more information, see Using Collective Attributes.
- **Super.** If this new attribute extends an existing attribute, enter or select the name of the existing super type.
- **Origin.** Enter the source of this new attribute type, for example, RFC 4512.

  To view the source of all the schema elements in the directory, select **Show All** from the **View** menu.
- **Schema File Extension.** If the attribute type's definition is contained in a file, enter the path to the file.

6. Click **Create** to create the new attribute.

33.6.2 Adding an Attribute Based on an Existing Attribute

Use Oracle Unified Directory Services Manager to add an attribute type that is based on an existing attribute type.

To add an attribute type that is based on an existing attribute type:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. Select the attribute on which you want to base the new attribute type.
5. Click the **Create like** icon.
6. Certain fields are completed by default, based on the attribute that you selected.

   Complete the remaining fields for the new attribute type.

   For information about the fields and their values, see Adding a New Attribute Type.
7. Click **Create** to create the new attribute.

33.6.3 Modifying an Attribute

Use Oracle Unified Directory Services Manager to modify an existing attribute type.

To modify an existing attribute type:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.

3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. Select the attribute type that you want to modify.
5. Modify the required fields, on the right hand pane.

   For information about the fields, see Adding a New Attribute Type.
6. Click **Apply** to save your changes.

33.6.4 Deleting an Attribute

Use Oracle Unified Directory Services Manager to delete an existing attribute type.

To delete an existing attribute type:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. Select the attribute type that you want to delete.
5. Click the **Delete** icon and click **OK** to confirm the deletion.
6. Click **Apply** to save your changes.
7. Click the **Refresh** icon to refresh the list of attributes on the left hand pane and confirm that the attribute has been deleted from the schema.

> **Note:**
>
> The server will return an error if you attempt to delete an attribute type that is already referenced by one or more entries in the server.

33.6.5 Viewing All Directory Attributes

Use Oracle Unified Directory Services Manager to view all existing attribute types.

To view all existing attribute types:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. All the attributes that are defined in the schema are listed in the left hand pane.
5. Select an attribute to display its properties in the right hand pane.

33.6.6 Searching for Attributes

Use Oracle Unified Directory Services Manager to search for a specific attribute types.

To search for a specific attribute types:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. All the attributes that are defined in the schema are listed in the left hand pane.
5. Enter part or all of the attribute name in the **Search** field and click the **Go** icon.

   The search field supports pattern matching. For example, enter `*uid` to find all attributes that end with the string `uid`.
6. Select an attribute to display its properties in the right hand pane.

33.6.7 Viewing the Indexing Details of an Attribute

Indexes are configured per server and index configuration is not replicated. A local database index is used to find entries that match search criteria. A VLV index is used to process searches efficiently with VLV controls. Unindexed searches are denied by default, unless the user has the unindexed-search privilege.

A local database index can be one of the following types:

- **approximate** - Improves the efficiency of searches using approximate search filters.
- **equality** - Improves the efficiency of searches using equality search filters.
- **ordering** - Improves the efficiency of searches using "greater than or equal to" or "less than or equal to" search filters.
- **presence** - Improves the efficiency of searches using presence search filters.
- **substring** - Improves the efficiency of searches using substring search filters.

To view the indexes that are defined for an attribute by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. The **Attributes** panel is expanded by default. If it is not expanded, click the arrow to expand it.
4. Select an attribute to display its properties in the right hand pane.
5. Scroll down to the **Indexed** property to view the indexing details for that attribute.

33.6.8 Adding a New Object Class

Use Oracle Unified Directory Services Manager (OUDSM) to add a new attribute type to the schema.

To add a new attribute type to the schema by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Click the **Object classes** panel to expand it.

   All existing object classes are displayed on the left pane.
4. Click the **Add** icon.
5. Complete the following information on the **Create new object class** window:
   - **Name.** Enter a unique name for the new object class.
   - **Object ID.** Specify the OID that uniquely identifies the object class in the directory server.Oracle Unified Directory supports the use of non-numeric OIDs for easy identification if the schema is used internally within the organization. However, for this release OUDSM supports numeric OIDs only.
   - **Description.** Enter a human-readable description of the object class.
   - **Type.** Specify the type of object class. Possible values are as follows:
     - **Structural.** A structural object class defines the core type for any entry that contains it. An entry must have exactly one structural class (although that structural class can inherit from other structural or abstract classes).
     - **Auxiliary.** An auxiliary object class does not define the core type of an entry, but defines additional characteristics of that entry. An entry can contain zero or more auxiliary object classes. The set of auxiliary classes that are allowed for use in an entry can be controlled by a DIT content rule that is associated with that entry's structural object class.
     - **Abstract.** An abstract object class cannot be used directly in an entry but must be subclassed by either a structural object class or an auxiliary object class. The subclasses inherit any required attribute type, optional attribute type, or both attribute types as defined by the abstract class.
   - **Superclass.** Click the **Add** icon to specify one or more superior object classes. The new object class will inherit elements from its superior object classes.
   - **Mandatory Attributes.** Click the **Add** icon to specify the set of attribute types that are required to be present (that is, have at least one value) in entries with that object class.
   - **Optional Attributes.** Click the **Add** icon to specify the set of attribute types that are allowed but not required to be present in entries with that object class.
   - **Inherited Attributes.** After the object class has been created, this field indicates the attributes that are inherited from the superior object classes of this object class.
   - **Origin.** Enter the source of this new object class, for example, RFC 4512.

     To view the source of all the schema elements in the directory, select **Show All** from the **View** menu.
   - **Schema File Extension.** If the definition of the new object class is contained in a file, enter the path to the file.
6. Click **Create** to create the new object class.

33.6.9 Adding an Object Class Based on an Existing Object Class

Use Oracle Unified Directory Services Manager to add an object class that is based on an existing object class.

To add an object class that is based on an existing object class:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Object classes** panel.
4. Select the object class on which you want to base the new object class.
5. Click the **Create like** icon.
6. Certain fields are completed by default, based on the object class that you selected. The existing object class is used as the superior object class for the new object class

   Complete the remaining fields for the new object class.

   For information about the fields and their values, see Adding a New Object Class.
7. Click **Create** to create the new object class.

33.6.10 Viewing the Properties of an Object Class

Use Oracle Unified Directory Services Manager to view the properties of an existing object class.

To view the properties of an existing object class:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Object Classes** panel.
4. All the object classes that are defined in the schema are listed in the left hand pane.
5. Select an object class to display its properties in the right hand pane.

33.6.11 Modifying an Object Class

Use Oracle Unified Directory Services Manager to modify an existing object class.

To modify an existing object class:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Object Classes** panel.
4. Select the object class that you want to modify.
5. Modify the required fields, on the right hand pane.

   For information about the fields, see Adding a New Object Class.
6. Click **Apply** to save your changes.

33.6.12 Deleting an Object Class

Use Oracle Unified Directory Services Manager to delete an existing object class.

To delete an existing object class:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Object Classes** panel.
4. Select the object class that you want to delete.
5. Click the **Delete** icon and click **OK** to confirm the deletion.
6. Click **Apply** to save your changes.
7. Click the **Refresh** icon to refresh the list of attributes on the left hand pane and confirm that the object class has been deleted from the schema.

> **Note:**
>
> The server will return an error if you attempt to delete an object class that is already referenced by one or more entries in the server.

33.6.13 Searching for Object Classes

Use Oracle Unified Directory Services Manager to search for a specific object class.

To search for a specific object class:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Object Classes** panel.
4. All the object classes that are defined in the schema are listed in the left hand pane.
5. Enter part or all of the object class name in the **Search** field and click the **Go** icon.

   The search field supports pattern matching. For example, enter `*person` to find all object classes that end with the string `person`.
6. Select an object class to display its properties in the right hand pane.

33.6.14 Displaying a List of LDAP Syntaxes

LDAP syntaxes are essentially data type definitions. The syntax for an attribute type indicates the type of data that should be held by the corresponding values. Syntaxes can be used to determine whether a particular value is acceptable for a given attribute, and to provide information about how the directory server should interact with existing values.

Oracle Unified Directory supports the ability to reject values that violate the associated attribute syntax, and this is the default behavior for the purposes of standards compliance. It is possible to disable attribute syntax checking completely if necessary. It is also possible to accept values

that violate the associated syntax but log a warning message to the directory server's error log when this occurs. For information about disabling schema checking, see Configuring Schema Checking.

You cannot modify the LDAP syntaxes but you can view all existing LDAP syntaxes.

To view all existing LDAP syntaxes by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Syntaxes** panel.
4. All the supported LDAP syntaxes are listed in the left hand pane.
5. Select a syntax to display its properties in the right hand pane.

   The information that is displayed includes all of the attributes and matching rules that currently refer to that syntax.

33.6.15 Searching for a Syntax

Use Oracle Unified Directory Services Manager to search for a specific LDAP syntax.

To search for a specific LDAP syntax:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Syntaxes** panel.
4. All the supported LDAP syntaxes are listed in the left hand pane.
5. Enter part or all of the syntax name in the **Search** field and click the **Go** icon.

   The search field supports pattern matching. For example, enter `*time` to find all syntaxes that end with the string `time`.
6. Select a syntax to display its properties in the right hand pane.

33.6.16 Displaying a List of LDAP Matching Rules

Matching rules are used by the directory server to compare two values for the same attribute, that is, to perform matching operations on them.

There are several different types of matching rules, including the following:

- **Equality matching rules.** These matching rules are used to determine whether two values are logically equal to each other. Different implementations of equality matching rules can use different criteria for making this determination (for example, whether to ignore differences in capitalization or deciding which spaces are significant).
- **Ordering matching rules.** These matching rules are used to determine the relative order for two values, for example, when evaluating greater-or-equal or less-or-equal searches, or when the results need to be sorted.
- **Substring matching rules.** These matching rules are used to determine whether a given substring assertion matches a particular value.

- **Approximate matching rules.** These matching rules are used to determine whether two values are approximately equal to each other. This is frequently based on "sounds like" or some other kind of fuzzy algorithm. Approximate matching rules are not part of the official LDAP specification, but they are included in Oracle Unified Directory for added flexibility.

You cannot modify the matching rules but you can view all existing matching rules.

To view all existing matching rules by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Matching Rules** panel.
4. All the configured matching rules are listed in the left hand pane.
5. Select a matching rule to display its properties in the right hand pane.

   The information that is displayed includes all of the attributes and matching rules that currently refer to that matching rule.

33.6.17 Searching for a Matching Rule

Use Oracle Unified Directory Services Manager to search for a specific matching rule.

To search for a specific matching rule:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Matching Rules** panel.
4. All the configured matching rules are listed in the left hand pane.
5. Enter part or all of the matching rule name in the **Search** field and click the **Go** icon.

   The search field supports pattern matching. For example, enter `*match` to find all matching rules that end with the string `match`.
6. Select a matching rule to display its properties in the right hand pane.

33.6.18 Displaying a List of Content Rules

Content rules provide a mechanism for defining the content that can appear in an entry. At most one content rule may be associated with an entry, based on its structural object class. If such a rule exists for an entry, then it will work with the object classes contained in that entry to define which attribute types must, may, and must not be present in the entry, as well as which auxiliary classes the entry may include.

To view all the content rules that are configure in the server by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. All the configured content rules are listed in the left hand pane.
5. Select a content rule to display its properties in the right hand pane.

33.6.19 Searching for a Content Rule

Use Oracle Unified Directory Services Manager to search for a specific content rule.

To search for a specific content rule:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. All the configured content rules are listed in the left hand pane.
5. Enter part or all of the content rule name in the **Search** field and click the **Go** icon.
6. Select a content rule to display its properties in the right hand pane.

33.6.20 Creating a New Content Rule

Use Oracle Unified Directory Services Manager to add a new content rules to the schema.

To add a new content rules to the schema:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. Click the **Add** icon.
5. Complete the following information on the **Create new content rule** window:
   - **Name.** Enter a unique name for the new content rule.
   - **Structural Object Class.** Specify the name of the structural object class with which this content rule is associated.
   - **Description.** Enter a human-readable description of the content rule.
   - **Auxiliary Object Classes.** Click the **Add** icon to specify the list of auxiliary object classes that may be present in entries with the associated structural class. If no values are provided, such entries will not be allowed to have any auxiliary object classes. You can specify the allowed auxiliary object classes by using their names or OIDs.
   - **Mandatory Attributes.** Click the **Add** icon to specify the list of attribute types that are required to be present in entries with the associated structural class. This list is in addition to the attribute types that are required by the object classes included in the entry. These additional attribute types do not need to be allowed by any of those object classes. You can specify the mandatory attributes by using their names or OIDs.
   - **Optional Attributes.** Click the **Add** icon to specify the list of attribute types that are allowed, but not required, to be present in entries with the associated structural class. This list is in addition to the attribute types that are allowed by the object classes included in the entry. You can specify the optional attributes by using their names or OIDs.
   - **Disallowed Attributes.** Click the **Add** icon to specify the list of attribute types that are prohibited from being present in entries with the associated structural class. This list may not include any attribute types that are required by the structural class or any of the allowed auxiliary classes. The list can be used to prevent the inclusion of attribute

types which would otherwise be allowed by one of those object classes. You can specify the disallowed attributes by using their names or OIDs.

- **Origin.** Enter the source of this new content rule, for example, RFC 4517.

  To view the source of all the schema elements in the directory, select **Show All** from the **View** menu.

- **Schema File Extension.** If the content rule's definition is contained in a file, enter the path to the file.

6. Click **Create** to create the new content rule.

33.6.21 Creating a Content Rule Based on an Existing Content Rule

Use Oracle Unified Directory Services Manager to add a content rule that is based on an existing content rule.

To add a content rule that is based on an existing content rule:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. Select the content rule on which you want to base the new content rule.
5. Click the **Create like** icon.
6. Certain fields are completed by default, based on the content rule that you selected.

   Complete the remaining fields for the new content rule.

   For information about the fields and their values, see Creating a New Content Rule.
7. Click **Create** to create the new content rule.

33.6.22 Modifying a Content Rule

Use Oracle Unified Directory Services Manager to modify an existing content rule.

To modify an existing content rule:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. Select the content rule that you want to modify.
5. Modify the required fields, on the right hand pane.

   For information about the fields, see Creating a New Content Rule.
6. Click **Apply** to save your changes.

33.6.23 Deleting a Content Rule

Use Oracle Unified Directory Services Manager to delete an existing content rule.

To delete an existing content rule:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Schema** tab.
3. Expand the **Content Rules** panel.
4. Select the content rule that you want to delete.
5. Click the **Delete** icon and click **OK** to confirm the deletion.
6. Click **Apply** to save your changes.
7. Click the **Refresh** icon to refresh the list of content rules on the left hand pane and confirm that the content rule has been deleted from the schema.

34

Moving from a Test to a Production Environment

You can move Oracle Unified Directory from a source environment, such as a test environment, to a target environment, such as a production environment. You can develop and test applications in a source environment, and then eventually roll out the test applications and, optionally, test data to your target environment.
Topics:

> **Note:**
>
> The Oracle Unified Directory "test to production" plug-in offers a subset of the functionality that is provided by the Oracle Fusion Middleware "test to production" framework.
>
> The information in this chapter is specific to Oracle Unified Directory. For a comprehensive description of moving other Fusion Middleware components between environments, see Moving from a Test to a Production Environment section in the *Administering Oracle Fusion Middleware*.

34.1 Introduction to Moving Across Environments

You can move Oracle Unified Directory from a source environment to a target environment. Moving an Oracle Unified Directory installation minimizes the amount of work that would otherwise be required to reapply all the customization and configuration changes made in one environment to another.

You can install, configure, customize, and validate Oracle Unified Directory in a test environment. Once the system is stable and performs as required, you can create the production environment by moving a copy of the server and its configuration from the test environment, instead of redoing all the changes that were incorporated into the test environment.If you have an existing production environment, you can move any modifications of the test environment, such as customization, to the production environment.

Moving an Oracle Unified Directory installation from a test to a production environment assumes that the production environment is on the same operating system as the test environment. In addition, the operating system architecture must be the same in both environments. For example, both environment must be running 32-bit operating systems or 64-bit operating systems.

34.2 Limitations in Moving from Test to Production

Certain limitations and restrictions apply when you move an Oracle Unified Directory installation between environments.

Note the following limitations and restrictions:

- Moving from a test to a production environment is supported for directory server instances only. You cannot move a proxy server instance or a replication gateway server instance between environments.
- You cannot move a replicated topology. To move an entire replicated topology, you must first move each server instance in the topology, then configure replication manually between the server instances. If you move a server that is part of a replicated topology, the replication configuration is removed from the configuration in the destination environment.
- Security data is not moved during the test to production process. This includes the following elements:
  - the SSL configuration (keystore, truststore, and other security configuration located in the `config` directory by default)
  - the SNMP V3 security file (located in the `config/snmp` directory by default)

34.3 Moving a Test to Production Environment

The move from a test to a production environment starts with the installation of Oracle Unified Directory binaries in the production system followed by the movement of existing configuration and data.

Follow these three steps to perform the move from a test to a production environment:

1. Moving the Oracle Unified Directory binaries to the production system. See Moving the Binaries.
2. Moving the Oracle Unified Directory configuration to the production system. See Moving the Configuration Between Environments.
3. Moving the data to the production system. See Moving the Data.

These procedures assume that you are moving an Oracle Unified Directory test system to a new production deployment (and do not have an existing production system).

34.3.1 Moving the Binaries

You must install the Oracle Unified Directory binaries in the production system to host Oracle Unified Directory in the Oracle home directory.

To obtain a copy of the Oracle Unified Directory binaries on the new production system, install the binaries as described in “Installing Oracle Unified Directory” in the *Installing Oracle Unified Directory*.

34.3.2 Moving the Configuration Between Environments

Use the `oudCopyConfig` command to copy an existing configuration from the source environment and the `oudPasteConfig` command to paste the copied configuration into the target environment.

Follow these steps to move the configuration between environments:

1. Copying the configuration from the source environment. See Copying the Configuration.
2. Editing the configuration, if required. See Editing the Configuration.
3. Pasting the configuration in the target environment. See Pasting the Configuration.

34.3.2.1 Copying the Configuration

To obtain a copy of an existing configuration, run the `oudCopyConfig` command in the source environment.

On UNIX systems, run the command as follows:

```
$ OUD_ORACLE_HOME/bin/oudCopyConfig -javaHome java_home \
  -sourceInstanceHomeLoc instance_dir -archiveLoc archive_location \
  -logDirLoc log_directory
```

For example:

```
$ OUD_ORACLE_HOME/bin/oudCopyConfig -javaHome /usr/jdk \
  -sourceInstanceHomeLoc /local/asinst_1 -archiveLoc /tmp/oud.jar \
  -logDirLoc /tmp/logs
```

On Windows systems, run the command as follows:

```
$ OUD_ORACLE_HOME\bat\oudCopyConfig.bat -javaHome java_home \
  -sourceInstanceHomeLoc instance_dir -archiveLoc archive_location \
  -logDirLoc log_directory
```

For a complete synopsis of the `oudCopyConfig` command, see oudCopyConfig.

The `oudCopyConfig` command performs the following actions:

- creates an archive (*archive_location*) that contains the required configuration data to move the test instance (*instance_dir*) to a production environment. `-archiveLoc` specifies the full path to the archive.
- creates a move plan in the archive.
- logs any messages to *log_directory*. If not specified, the default location of logged messages is the system temporary directory.

34.3.2.2 Editing the Configuration

You can modify certain configuration parameters by editing the *move plan*. A move plan is an XML file that exposes customizable parameters during the move across environments.

The move plan is generated when you run the `oudCopyConfig` command and is used by the `oudPasteConfig` command to duplicate the configuration.

After you have copied the configuration, edit the configuration as follows:

1. Run the `oudExtractMovePlan` command to obtain a copy of the configuration. On UNIX systems, run the command as follows:

   ```
   $ OUD_ORACLE_HOME/bin/oudExtractMovePlan -javaHome java_home \
     -archiveLoc archive_location -planDirLoc moveplan_dir \
     -logDirLoc log_directory
   ```

   For example:

```
$ OUD_ORACLE_HOME/bin/ExtractMovePlan -javaHome /usr/jdk \
  -archiveLoc /tmp/oud.jar -planDirLoc /tmp \
  -logDirLoc /tmp/logs
```

On Windows systems, run the command as follows:

```
$ OUD_ORACLE_HOME\bat\oudExtractMovePlan.bat -javaHome java_home \
  -archiveLoc archive_location -planDirLoc moveplan_dir \
  -logDirLoc log_directory
```

For a complete synopsis of the `oudextractMovePlan` command, see oudExtractMovePlan.

The `oudExtractMovePlan` command creates an editable version of the configuration in a file named `moveplan.xml`, in the location specified by the `-planDirLoc` argument. This directory must exist, and be writable.

2. In a text editor, edit the `moveplan.xml` file, as required.

   You can configure the following parameters in the move plan:

   - OUD non SSL port
   - OUD SSL port
   - OUD admin connector port
   - SNMP listen port
   - SNMP trap port
   - JMX port
   - OUD root user password file
   - SMTP server and port
   - Absolute paths to files or directories, including the following:
     - Backup directory
     - Database directory
     - Profile directory
     - Dictionary file
     - Referential integrity plug-in log file
     - SMTP account status notification handler message template file

3. Save the `moveplan.xml` file.

34.3.2.3 Pasting the Configuration

When you have edited the move plan, paste the configuration into the target environment as follows:

1. Move the archive and move plan to the target host.

   In most scenarios, the test environment and the production environment are on separate machines. You must therefore move or copy the archive and move plan to the target machine.

   If your test and production environments are on the same machine, this step is unnecessary.

2. Paste the configuration in the target environment, by running the `oudPasteConfig` command on the target environment.

On UNIX systems, run the command as follows:

```
$ OUD_ORACLE_HOME/bin/oudPasteConfig -javaHome java_home \
  -targetInstanceHomeLoc instance_dir -archiveLoc archive_location \
  -targetOracleHomeLoc ORACLE_HOME -movePlanLoc move_plan_location \
  -logDirLoc log_directory -targetInstanceName instance_name
```

For example:

```
$ OUD_ORACLE_HOME/bin/oudPasteConfig -javaHome /usr/jdk \
  -targetInstanceHomeLoc /local/asinst_2 -archiveLoc /tmp/oud.jar \
  -targetOracleHomeLoc /local/ORACLE_HOME -movePlanLoc /tmp/moveplan.xml \
  -logDirLoc /tmp/logs -targetInstanceName asinst_2
```

On Windows systems, run the command as follows:

```
$ OUD_ORACLE_HOME\bat\oudPasteConfig.bat -javaHome java_home \
  -targetInstanceHomeLoc instance_dir -archiveLoc archive_location \
  -targetOracleHomeLoc ORACLE_HOME -movePlanLoc move_plan_location \
  -logDirLoc log_directory -targetInstanceName instance_name
```

For a complete synopsis of the `oudPasteConfig` command, see oudPasteConfig.

The `oudPasteConfig` command creates a new server instance with the configuration obtained from the archive and the amended move plan, if any.

34.3.3 Moving the Data

The simplest way to move data from a test system to a production system is to export the data from the test system, and import it to the production system.

For information on how to do this, see Importing and Exporting Data.

Part VII

Advanced Administration: Monitoring and Tuning Performance

It is important to know how to monitor Oracle Unified Directory, how to manage log files to assist in monitoring system activity, and how to significantly improve performance through some basic tuning.

Topics:

35
Monitoring Oracle Unified Directory

Oracle Unified Directory provides an extensible monitoring framework to view the statistics on a server instance or on a replicated topology.
Topics:

35.1 Overview of Monitoring Information

You can use logs and alerts for monitoring information and performance data. You can also monitor the server using LDAP and SNMP.

- logs

  For information about configuring logs, see Configuring Logs.
- alerts

  For information about configuring alerts, see Configuring Alerts and Account Status Notification Handlers.
- `cn=monitor`

  For information about `cn=monitor`, see Monitoring the Server with LDAP.
- `DIRECTORY_SERVER_MIB`, defined by RFC 2605

  For information about monitoring the server with SNMP, see Monitoring the Server With SNMP.

To access the monitoring information, ensure that you have the required protocol:

- For logs you need a file system.
- For alerts you need JMX:RMI or SMTP.
- For `cn=monitor` you need LDAP or JMX:RMI (for example jconsole).
- For DIRECTORY_SERVER_MIB you need SNMP.

35.2 Configuring Monitor Providers

Monitor providers are enabled by default and provide information about the server that can be useful for monitoring or troubleshooting purposes. The `cn=monitor` entry contains the monitoring information that is published by the monitor providers.

When the monitor provider is disabled, the provided information is no longer available under `cn=monitor`.

You can configure monitor providers using the `dsconfig` command. For more information, see Managing the Server Configuration Using `dsconfig`.

This section includes the following topics:

- Viewing Monitor Providers.
- Disabling Monitor Providers.

35.2.1 Viewing Monitor Providers

Use the `dsconfig` command to view the list of existing monitor providers.

Run the `dsconfig` command with the `list-monitor-providers` subcommand, as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-monitor-providers

Monitor Provider   : Type              : enabled
-------------------:-------------------:--------
Client Connections : client-connection : true
Entry Caches       : entry-cache       : true
JVM Memory Usage   : memory-usage      : true
JVM Stack Trace    : stack-trace       : true
System Info        : system-info       : true
Version            : version           : true
```

35.2.2 Disabling Monitor Providers

Use the `dsconfig` command with `set-monitor-provider-prop` to disable a monitor provider.

For example, to set the JVM Stack Trace monitor provider to false, use the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-monitor-provider-prop --provider-name "JVM Stack Trace" \
  --set enabled:false
```

Running the `dsconfig` command with the `list-monitor-providers` subcommand now shows the JVM Stack Trace monitor provider as false:

```
Monitor Provider   : Type              : enabled
-------------------:-------------------:--------
Client Connections : client-connection : true
Entry Caches       : entry-cache       : true
JVM Memory Usage   : memory-usage      : true
JVM Stack Trace    : stack-trace       : false
System Info        : system-info       : true
Version            : version           : true
```

35.3 Configuring Logs

Oracle Unified Directory provides several types of logs: access logs, audit logs, error logs, debug logs, and a replication repair log. The replication repair log is read-only and its use is restricted to enabling replication conflict resolution.

The following topics describe how to configure access, audit, error, and debug logs by using the `dsconfig` command-line interface or Oracle Unified Directory Services Manager:

- Configuring Logs Using `dsconfig`
- Configuring Logs Using OUDSM
- Logging Operations to Access Log Publishers
- Masking Attributes in the Audit Log

In addition, the section describes how to log admin operations.

For a breakdown of the result codes found in the logs, see result code.

35.3.1 Configuring Logs Using `dsconfig`

The easiest way to configure logging with `dsconfig` is to use the command in interactive mode, which walks you through the configuration.

This section provides the required commands in non-interactive mode, so that you can see the specific parameters that are set. For more information about `dsconfig`, see Managing the Server Configuration Using `dsconfig`.

Log configuration includes the definition of three configuration objects:

- **Log publisher.** A log publisher is defined for each logger. The log publisher type corresponds to the type of log. For more information about log publishers, see Configuring Log Publishers.
- **Log retention policy.** The retention policy determines how long archived log files are stored. For more information about log retention policies, see Configuring Log Retention Policies.
- **Log rotation policy.** The rotation policy determines how often log files are rotated. For more information on log rotation policies, see Configuring Log Rotation Policies.
- Configuring Logs for HTTP/HTTPS Operations

35.3.1.1 Configuring Log Publishers

Oracle Unified Directory provides several log publishers by default.

Any number of log publishers of any type can be defined and active at any time. This means that you can log to different locations or different types of repositories and that you can specify various sets of criteria for what to include in the logs.

For more information about the configuration properties associated with log publishers, see *Configuration Reference for Oracle Unified Directory.*

This section includes the following topics:

- Viewing Existing Log Publishers
- Enabling a Log Publisher

- Deleting a Log Publisher
- Logging in ODL Format
- Logging Internal Operations
- Logging Additional Connection Details
- Configuring the Name of Rotated Log Files Using Local Time Stamp

35.3.1.1.1 Viewing Existing Log Publishers

To view the existing log publishers using the `dsconfig` command:

1. To view the existing log publishers run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-log-publishers
```

The default output will be similar to the following:

```
Log Publisher                          Type                           enabled
-------------------------------------------------------------------------------
File-Based Access Control Logger    : file-based-access-control    : false
File-Based Access Logger            : file-based-access            : false
File-Based Admin Logger             : file-based-access            : false
File-Based Audit Logger             : file-based-audit             : false
File-Based Debug Logger             : file-based-debug             : false
File-Based Error Logger             : file-based-error             : true
File-Based High eTime Access Logger : high-etime-file-based-access : false
File-Based HTTP Admin Logger        : file-based-http-access       : false
File-Based HTTP NSCA Access Logger  : file-based-http-access       : false
File-Based HTTP W3C Access Logger   : file-based-http-access       : false
High eTime Oracle Access Logger     : high-etime-file-based-access : false
Oracle Access Logger                : file-based-access            : false
Oracle Admin Access Logger          : file-based-access            : false
Oracle Error Logger                 : file-based-error             : false
Replication Repair Logger           : file-based-error             : true
```

2. To display the properties of a log publisher run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-log-publisher-prop --publisher-name "File-Based Error Logger"
```

35.3.1.1.2 Enabling a Log Publisher

Not all of the log publishers are enabled by default. If a log publisher is disabled, messages of that type are not logged.

To enable a log publisher, set its enabled property to true. For example, to enable the audit logger, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-log-publisher-prop --publisher-name "File-Based Audit Logger" \
  --set enabled:true
```

When a log publisher is enabled, the server immediately starts logging messages to the appropriate publisher. You do not need to restart the server for this change to take effect.

35.3.1.1.3 Deleting a Log Publisher

You use the `dsconfig` command to delete an existing log publisher.

To delete a log publisher, for example the `File-Based Audit Logger` run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
delete-log-publisher --publisher-name "File-Based Audit Logger"
```

The logger is deleted successfully.

> **Note:**
>
> The audit logger is a `File-Based Access Log`. Therefore, to create a `File-Based Audit Logger`, you must run the following `dsconfig` command using the `advanced` option in the interactive mode to set the Java class as `org.opends.server.loggers.TextAuditLogPublisher`.
>
> ```
> $ dsconfig -X -j pwd-file --advanced
> ```
>
> Alternatively, you can also create an audit logger using the non-interactive mode of the `dsconfig` command as follows:
>
> ```
> dsconfig create-log-publisher \
>          --set enabled:false \
>          --set log-file:log/myauditlog \
>          --set java-class:org.opends.server.loggers.TextAuditLogPublisher \
>          --type file-based-access \
>          --publisher-name myauditlog \
>          --hostname localhost \
>          --port 4444 \
>          --trustAll \
>          --bindDN cn=Directory Manager \
>          --bindPasswordFile /tmp/password.txt \
>          --no-prompt
> ```

35.3.1.1.4 Logging in ODL Format

Oracle Unified Directory also writes diagnostic log files in the Oracle Diagnostic Logging (ODL) format.

ODL is disabled by default. To enable ODL, set the `enabled` property of the ODL Access Log publisher or the ODL Error Log publisher to `true`. The following example enables the access logger:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  set-log-publisher-prop --publisher-name "Oracle Access Logger" \
  --set enabled:true
```

To enable the error logger, use `--publisher-name "Oracle Error Logger"`.

ODL access logs are stored in the following file:

```
instance_dir/OUD/logs/access.log
```

ODL error logs are stored in the following directory:

```
instance_dir/OUD/logs/errors.log
```

> **Note:**
>
> After enabling the ODL loggers, you should disable the standard file-based loggers unless you specifically want to maintain logs in both formats. Any write to a file is a costly operation and affects performance.

For more information about ODL, including an explanation of the log file format, see Managing Log Files and Diagnostic Data in the *Oracle Fusion Middleware Administrator's Guide*.

35.3.1.1.5 Logging Internal Operations

You can log internal operations in versions 11.1.2.3 and above by setting `add operations-to-log` property to `internal`.

In versions 11.1.2.2 and below, you could log internal operations by setting the value of `suppress-internal-logging` property for log publishers to `false`. From 11.1.2.3 version onwards, `suppress-internal-logging` property has been deprecated. You can now use `add operations-to-log` property to log internal operations (such as operations performed by the LDIF connection handler and certain plug-ins). By default, this property is set to `internal`. When the value of the `add operations-to-log` property is `internal`, it will automatically log the internal operations.

The following example sets the `add operations-to-log` property to `internal` for the file-based access logger:

```
dsconfig set-log-publisher-prop \
--publisher-name File-Based\ Access\ Logger \
--add operations-to-log:internal \
--hostname localhost \
--port 4444 \
-X \
--bindDN cn=directory\ manager \
--bindPasswordFile /tmp/password \
--no-prompt
```

35.3.1.1.6 Logging Additional Connection Details

When the `log-connection-details` flag is set to `true`, log messages for LDAP operations will have additional details such as bindDN, protocol, and client and server IP addresses.

If the connection is made over a secure channel, the TLS version and the negotiated cipher suite will also get logged.

> **Note:**
>
> You can now log SSL connection information, which provides you the flexibility to effectively analyze and debug SSL issues.

The following command describes how to set the `log-connection-details` flag to `true` for file-based access logger.

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
set-log-publisher-prop --publisher-name "File-Based Access Logger" \
--set log-connection-details:true
```

35.3.1.1.7 Configuring the Name of Rotated Log Files Using Local Time Stamp

By default, Oracle Unified Directory automatically renames (rotates) its local server log file using date stamp in GMT format.

You can change these default settings for log file rotation. You can configure a server instance to include a local time stamp in the file name of rotated log files.

To configure the log file names using local time stamp, you must set the `log-file-use-local-time` property of the appropriate log publisher to `true`. The following example describes how to set up the local time stamp in the file name of access rotated log files:

```
dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
set-log-publisher-prop --publisher-name "File-Based Access Logger" \
--set log-file-use-local-time:true
```

> **Note:**
>
> The rotated log file name using local time stamp follows the format used by Oracle Directory Server Enterprise Edition to ensure compatibility.

35.3.1.2 Configuring Log Retention Policies

Log retention policies dictate size and space limits for log files.Oracle Unified Directory provides the following three log retention policies:

- **File count retention** (`file-count`). By default, this policy sets the maximum number of log files to 10, for a specified type of log file.
- **Free disk space retention** (`free-disk-space`). By default, this policy sets a minimum remaining free disk space limit to 500 Mb, for a specified type of log file.
- **Size limit retention** (`size-limit`). By default, this policy sets the disk spaced used to a maximum of 500 Mb, for a specified type of log file.

By default, the log retention policy that is enabled is File count retention.

You can also create your own custom log retention policies. For more information, see Creating a Log Retention Policy.

This section contains the following topics:

- Viewing the Log Retention Policies
- Creating a Log Retention Policy
- Modifying a Log Retention Policy

35.3.1.2.1 Viewing the Log Retention Policies

You use the `dsconfig` command to view the existing log retention policies.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-log-retention-policies
```

The default output will be similar to the following:

```
Log Retention Policy             : Type            : disk-space-used : free-disk-space : number-of-files
---------------------------------:-----------------:-----------------:-----------------:----------------
File Count Retention Policy      : file-count      : -               : -               : 10
Free Disk Space Retention Policy : free-disk-space : -               : 500 mb          : -
Size Limit Retention Policy      : size-limit      : 500 mb          : -               : -
```

To list the log retention policy properties run the following `dsconfig` command

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-log-retention-policy-prop --policy-name "Free Disk Space Retention Policy"
```

35.3.1.2.2 Creating a Log Retention Policy

You use the `dsconfig` command to create a log retention policy.

Run the command as follows to create a log retention policy and to set it as enabled:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -w pwd-file -X -n
  create-log-retention-policy --policy-name MyMaxDiskSpace \
  --type size-limit --set disk-space-used:100mb
```

35.3.1.2.3 Modifying a Log Retention Policy

You use the `dsconfig` command to modify an existing log retention policy.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -w pwd-file -X -n \
  set-log-retention-policy-prop --policy-name "File Count Retention Policy" \
  --set number-of-files:20
```

Instead of setting a property value, you can add, reset or remove a property value, using the `--add`, `--reset`, or `--remove` subcommands instead of the `--set` subcommand. For details, see dsconfig

35.3.1.3 Configuring Log Rotation Policies

Log rotation policies dictate how often the files are rotated; or, how long to keep log files based on various criteria.

This section contains the following topics:

- Overview of Log Rotation Policies
- Viewing the Log Rotation Policies
- Creating a Log Rotation Policy

- Setting Log Rotation or Retention for a Specific Log File

35.3.1.3.1 Overview of Log Rotation Policies

Oracle Unified Directory provides the following four log rotation policies:

- **24 Hours time limit rotation policy**. By default, this policy sets the rotation interval to one day. You can configure the time of day.
- **7 Days time limit rotation policy**. By default, this policy sets the rotation interval to one week. You can configure the time of day.
- **Fixed time limit rotation policy**. By default, this policy sets the time of day that log files are to be rotated, to one minute before midnight.
- **Size time limit rotation policy**. By default, this policy sets a maximum size that log files can reach to 100 Mb, before the log file is rotated.

> **Note:**
>
> When multiple rotation policies are specified for the same log, the first threshold that is reached triggers the rotation.

The type of log rotation policy enabled by default depends on the log type.

- For access and audit logs, the following are enabled:
  - 24 Hours time limit rotation policy
  - Size time limit rotation policy
- For error and replication repair logs, the following are enabled:
  - 7 Days time limit rotation policy
  - Size time limit rotation policy

You can create your own custom log rotation policies.

35.3.1.3.2 Viewing the Log Rotation Policies

You use the `dsconfig` command to view an existing log rotation policy.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  list-log-rotation-policies
```

The default output will be similar to the following:

```
Log Rotation Policy                : Type       : file-size-limit : rotation-interval : time-of-day
-----------------------------------:------------:-----------------:-------------------:------------
24 Hours Time Limit Rotation Policy : time-limit : -               : 1 d               : -
7 Days Time Limit Rotation Policy   : time-limit : -               : 1 w               : -
Fixed Time Rotation Policy          : fixed-time : -               : -                 : 2359
Size Limit Rotation Policy          : size-limit : 100 mb          : -                 : -
```

To display the log rotation policy properties, run the following command:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  get-log-rotation-policy-prop "Fixed Time Rotation Policy"
```

35.3.1.3.3 Creating a Log Rotation Policy

You use the `dsconfig` command to create a log rotation policy.

The policy type can be one of the following:

- `size-limit`
- `fixed-time`
- `time-limit`

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -X -n \
  create-log-rotation-policy --policy-name my2DayPolicy \
  --type time-limit --set rotation-interval:2d
```

35.3.1.3.4 Setting Log Rotation or Retention for a Specific Log File

To set a rotation or retention policy on a specific log file, you must create a log publisher and set the log rotation or log retention policy.

Run the `dsconfig` command to set log rotation or retention for a specific log file:

```
$ dsconfig -h localhost -p 1444 -D "cn=Directoy manager" -j pwd-file -n -X \
  create-log-publisher --publisher-name myPublisher \
  --type file-based-access --set log-file:logs/myLogs --set enabled:true \
  --set retention-policy:MyMaxDiskSpace --set rotation-policy:my2DayPolicy
```

35.3.1.4 Configuring Logs for HTTP/HTTPS Operations

The following topics describe the different types of loggers that you can configure for HTTP/HTTPS operations:

- Overview of Access Logger for SCIM and REST Operations
- Overview of Admin Access Logger

35.3.1.4.1 Overview of Access Logger for SCIM and REST Operations

OUD provides detailed HTTP log publisher in two different log formats, World Wide Web Consortium (W3C) and Nagios Service Check Acceptor (NSCA) to capture the request and response details for SCIM and REST operations.

The following HTTP Access loggers help you capture SCIM and REST request and response details:

> **Note:**
>
> When you set the `log-connection-details` flag to `true`, then the TLS version and the negotiated cipher suite are also captured along with appropriate error message for failed scenarios as part of `x-additional-details` column of logs.

- **HTTP W3C Logger:**
  For every HTTP operation, W3C format log contains values for respective fields (separated by space) in the following order:

  ```
  date c-ip cs-username s-ip cs-method cs-uri-stem cs-uri-query sc-status sc-
  bytes cs-bytes time-taken cs-version cs(User-Agent) cs(Cookie) cs(Referer)
  x-additional-details
  ```

  The following command describes how to set the `log-connection-details` flag to `true` for W3C logger:

  ```
  dsconfig -h localhost -p 8081 -D "cn=Directory Manager" -j pwd-file -X -n
  \
  set-log-publisher-prop --publisher-name "File-Based HTTP W3C Access
  Logger" \
  --set "log-connection-details:true"
  ```

- **HTTP NSCA Logger:**
  For every HTTP operation, NSCA format log contains values for respective fields (separated by space) in the following order:

  ```
  remote-host user-identifier auth-user request date status bytes referrer
  user-agent cookie connection-id etime x-additional-details
  ```

  The following command describes how to set the `log-connection-details` flag to `true` for NSCA logger:

  ```
  dsconfig -h localhost -p 8081 -D "cn=Directory Manager" -j pwd-file -X -n
  \
  set-log-publisher-prop --publisher-name "File-Based HTTP NSCA Access
  Logger" \
  --set "log-connection-details:true"
  ```

35.3.1.4.2 Overview of Admin Access Logger

For administration operations performed over HTTP, access log message contains information according to the value (**w3c** or **nsca**) configured for `access-log-format-mode` property for file-based HTTP admin logger.

> **Note:**
>
> When you set the `log-connection-details` flag to `true`, then log messages for admin operations will have additional details such as cipher suite, TLS version, and appropriate error messages.

The following command describes how to set the `log-connection-details` flag to `true` for file-based HTTP admin logger:

```
dsconfig -h localhost -p 8444 -D "cn=Directory Manager" -j pwd-file -X -n \
set-log-publisher-prop --publisher-name "File-Based HTTP Admin Logger" \
--set "log-connection-details:true"
```

> **Note:**
>
> The Admin logger supports the same format for W3C and NSCA as defined in Overview of Access Logger for SCIM and REST Operations.

35.3.2 Configuring Logs Using OUDSM

Use Oracle Unified Directory Services Manager (OUDSM) to configure logger properties, log rotation policies and log retention policies.

The following topics describe how to configure logs using OUDSM:

- Modifying Logger Properties
- Modifying Log Rotation Policies
- Modifying Log Retention Policies

35.3.2.1 Modifying Logger Properties

Oracle Unified Directory provides several log publishers, or loggers, by default. Any number of loggers of any type can be defined and active at any time. This means that you can log to different locations or different types of repositories and that you can specify various sets of criteria for what to include in the logs.

You cannot create a new log publisher with OUDSM, but you can modify the properties of an existing log publisher.

To configure logger properties by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Logging** element.
5. Expand the **Loggers** element and click the logger whose properties you want to modify.

   The properties of the logger are displayed in the right hand pane. The configurable properties will depend on the type of logger that you have selected. For a comprehensive list of all configurable properties and their allowed values, see the *Configuration Reference for Oracle Unified Directory*.

   Oracle Unified Directory provides the following general configuration policies depending on the type of logger you have selected:

   - **Enabled.** It indicates whether the Log Publisher is enabled for use.
   - **Log Publisher File Location.** It specifies the file name to use for the log files generated by the File-Based Access Log Publisher. The path to the file is relative to the server root.
   - **Log Publisher Permissions.** It indicates the UNIX permissions of the log files created by this File-Based Access Log Publisher.
   - **Operations to Log.** It indicates which operations must be logged.

     This property is only available for the access and audit log publishers.

- **Log Request and Response Controls.** It indicates whether the request controls and response controls should be logged along with the operations that are requested by the client applications.

  This property is only available for the access and audit log publishers.

- **Time Zone in Rotated Log File Names**. It indicates whether the local time of the server or Greenwich Mean Time (GMT) should be used in the rotated log file names.
- **Default Severity.** It specifies the default severity levels for the logger.

  This property is only available for the error log publishers.

- **Default Debug Level.** It specifies the lowest severity level of debug messages to log when none of the defined targets match the message.

  This property is only available for the debug log publishers.

For a comprehensive list of all configurable properties and their allowed values for each logger, see the *Configuration Reference for Oracle Unified Directory*.

> **Note:**
>
> You can configure the log rotation and log retention policies for the logger that you select in Step 5. For more information about configuring log rotation and log retention policies, see Modifying Log Rotation Policies and Modifying Log Retention Policies.

35.3.2.2 Modifying Log Rotation Policies

Log rotation policies dictate how often log files are rotated, that is to say, how long log files are kept based on various criteria.

Oracle Unified Directory provides the following four log rotation policies:

- 24 Hours time limit rotation policy. By default, this policy sets the rotation interval to one day. You can configure the time of day.
- 7 Days time limit rotation policy. By default, this policy sets the rotation interval to one week. You can configure the time of day.
- Fixed time limit rotation policy. By default, this policy sets the time of day that log files are to be rotated, to one minute before midnight.
- Size time limit rotation policy. By default, this policy sets a maximum size that log files can reach to 100 Mb, before the log file is rotated.

The type of log rotation policy that is enabled by default depends on the logger type.

To configure log rotation policies by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Logging** element.
5. Select the **Rotation Policies** element and modify the required properties.

You can also add a new rotation policy or delete an existing rotation policy by clicking the Add or Delete icons on this page, and completing the required information.

35.3.2.3 Modifying Log Retention Policies

Log retention policies dictate size and space limits for log files.Oracle Unified Directory provides the following three log retention policies by default:

- File count retention (file-count). By default, this policy sets the maximum number of log files to 10, for a specified type of log file.
- Free disk space retention (free-disk-space). By default, this policy sets a minimum remaining free disk space limit to 500 Mb, for a specified type of log file.
- Size limit retention (size-limit). By default, this policy sets the disk spaced used to a maximum of 500 Mb, for a specified type of log file. By default, the log retention policy enabled is File count retention.

To configure log retention policies by using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Logging** element.
5. Select the **Retention Policies** element and modify the required properties.

You can also add a new retention policy or delete an existing retention policy by clicking the Add or Delete icons on this page, and completing the required information.

35.3.3 Logging Operations to Access Log Publishers

Oracle Unified Directory provides a new parameter, admin logger, to specify the operations to log and how to configure logged operations in access log publishers.

- Overview of the Admin Logger
- Configuring Logged Operations in Access Log Publishers Using OUDSM

35.3.3.1 Overview of the Admin Logger

Oracle Unified Directory provides a mechanism for separating admin logs from user logs by means of Admin connector. Administration operations are now logged into a separate file that provides logging information associated with the administration traffic.

> **Note:**
>
> By default, Oracle Unified Directory supports a dedicated access logger, named the `File-Based Admin Access Logger`, which contains only operations of the administrator connector. Therefore, you do not have to perform any action specific action to log administration operations into a separate file.

You can configure the access logs to specify the type of operation to log using `operations-to-log` property. This property is optional, and has the following configurable values:

- SYNCHRONIZATION
- INTERNAL
- ADMINISTRATION
- USER
- ADMIN_BROWSING
- ALL

In that sense,Oracle Unified Directory supports the following operation types:

- Synchronization Operations

  Synchronization operations, such as locks, process synchronization, attribute mapping and transformation.

- Internal Operations

  Internal operations are internal, because they are initiated not by external requests from clients, but instead internally by plug-ins. You must use internal operation calls when the plug-in needs Directory Server to perform an operation for which no client request exists.

- Administration Operations

  Administration operations are performed on the admin network group, excluding operations associated with **network group selection** control.

- User Operations

  User operations are performed on any user network group, excluding operations associated with **network group selection** control.

- Admin Browsing Operations

  Admin browsing operations are associated with the network group selection control. This excludes operations associated with network group dependency.

> **Note:**
>
> Operations handled by network group that are created by a user and accessing admin suffixes is considered as User operations.

35.3.3.2 Configuring Logged Operations in Access Log Publishers Using OUDSM

Oracle Unified Directory Services Manager (OUDSM) groups the log publisher properties into the three different headers, based on the nature and behavior of the property:

- Logger General Properties
- Rotation and Retention Properties
- Advanced Properties

The Logger General Properties region is visible by default for all loggers and allows you to configure operations to log for file-based access loggers.

To configure operations to log in Access Log Publishers:

1. Connect to the directory server or directory proxy server from OUDSM, as described in Connecting to the Server Using OUDSM.

2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Logging** element.
5. Expand the **Loggers** element.
6. Click the file-based access logger that you want to modify, for instance **File-Based Admin Access Logger.**
7. In the **Logger General Properties** region, perform the following step:

   From the **Operation to Log** list, select the operations to log.
8. Click **Apply.**

35.3.4 Masking Attributes in the Audit Log

Oracle Unified Directory allows you to mask certain attributes in the audit log.

This section describes the following topics:

- Overview of Masking Attributes in the Audit Log
- Configuring Audit Log Masking

35.3.4.1 Overview of Masking Attributes in the Audit Log

Oracle Unified Directory enables you to control how certain attributes, such as `userpassword`, are displayed in the audit log.

By default,Oracle Unified Directory*masks* the following attributes in the audit log using a five-asterisk string (*********) so there are no discernible values. Unmasked attributes are displayed in the clear — unless they are an encrypted attribute or a password.

- Password attributes defined in the server
- Attributes defined as encrypted
- User-specified list of attributes to be masked in the audit log

> **Note:**
>
> Attribute masking is relevant only when the audit log is enabled. The audit log file is located at:
>
> `<OUD_INSTALLATION_PATH>/OUD/logs/audit`

Table 35-1 describes the parameters that control how password, encrypted, and user-specified attributes are displayed in the audit log.

Table 35-1 Audit Log Masking Configuration Parameters

Name	Format	Default Value	Single/ Multi-Valued	Optional	Description
`mask-passwords`	String representing a boolean (`true`/ `false`).	true	S	Yes	Enables or disables password masking. • **`true`** (default): Mask all passwords in the audit log. • **`false`**: Display passwords using their hashed value.
`masking-uses-encryption-config`	String representing a boolean (`true`/ `false`).	true	S	Yes	Enables or disables the data encryption configuration that determines which attribute and suffix values are masked in the audit log. • **`true`** (default): Use `attribute-encryption-include` and `encrypted-suffix` to determine which attributes to mask, if the attributes are in the defined suffix list. • **`false`**: Do not use data encryption configuration. **Note**: For information about the `attribute-encryption-include` and `encrypted-suffix` parameters, see Attribute Encryption Configuration Parameters. Whether this parameter is `true` or `false`, `masked-attribute` and `masked-suffix` are always operational.
`masked-attribute`	String representing a single attribute name or OID.	None	M	Yes	Use to define a list of attributes to mask. • Mask every attribute defined in this list. • Mask all attributes in all suffixes or, if you defined a list of suffixes using `masked-suffix`, then only mask the suffixes in that list. This parameter always uses the attributes defined in this list, regardless of the `masking-uses-encryption-config` value.
`masked-suffix`	String representing a single suffix	None	M	Yes	Use to define a list of suffixes to mask. • If you defined a list of suffixes using `masked-suffix`, then mask the entry attributes in the defined list of suffixes. • If the suffix list is empty, then mask the defined attributes in all suffixes. This parameter always uses the attributes defined in this list, regardless of the `masking-uses-encryption-config` value.

You can use standard `dsconfig` commands or `dsconfig` in interactive mode to read and modify these parameters. The easiest method to use is `dsconfig` in interactive mode, which functions like a wizard. Because interactive mode is self-explanatory, this section does not provide instructions for modifying the audit log configuration using interactive mode, but instead provides the equivalent `dsconfig` commands.

> **Note:**
>
> For more information about using `dsconfig`, see Using the `dsconfig` Command and Using `dsconfig` in Interactive Mode.

35.3.4.2 Configuring Audit Log Masking

You use `dsconfig` command to configure the audit log masking.

Run the command as follows:

```
./dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager"
    -j /security/password set-log-publisher-prop --publisher-name
    "File-Based Audit Logger" --set "maskpasswords:true"
./dsconfig -n -X -h localhost -p 1444 -D "cn=Directory Manager"
    -j /security/password get-log-publisher-prop --publisher-name
    "File-Based Audit Logger"

Property                : Value(s)
------------------------------:-----------------------------------------------
append                  : true
enabled                 : false
log-file                : logs/audit
log-file-permissions    : 640
log-file-use-local-time : false
mask-passwords          : true
masked-attribute        : -
masked-suffix           : -
masking-uses-encryption-config : true
operations-to-log       : adminbrowsing, administration,
                        : synchronization, user
retention-policy        : File Count Retention Policy
rotation-policy         : 24 Hours Time Limit Rotation Policy, Size
                        : Limit Rotation Policy
```

> **Note:**
>
> Configuration changes immediately take effect, but they are not retroactive. Updating the audit log configuration entry only affects *future* logs in the audit log file.

35.4 Configuring Alerts and Account Status Notification Handlers

Oracle Unified Directory provides mechanisms for transmitting alert and account status notifications by means of JMX extensions or SMTP extensions. You can configure the directory server to send alert notifications when an event occurs during processing.

Typical server events include server starts and shut downs, or problems that are detected by the server, such as an attempt to write to the configuration file.

You can also receive account status notifications when an event occurs during password policy processing, such as when accounts are locked out, accounts expire, passwords expire, and so on.

Alerts and account status notification handlers are configured by using the `dsconfig` command. For more information, see Managing the Server Configuration Using `dsconfig`.

For additional information about the topics in this section, see Managing Password Policies and "The Alert Handler Configuration" in the *Configuration Reference for Oracle Unified Directory.*

This section contains the following topics:

- Managing Alert Handlers
- Managing Account Status Notification Handlers

35.4.1 Managing Alert Handlers

Oracle Unified Directory provides mechanisms for transmitting alert and account status notifications by means of JMX extensions or SMTP extensions.

You can configureOracle Unified Directory to send alert notifications when an event occurs during processing. Typical server events include server starts and shut downs, or problems that are detected by the server, such as an attempt to write to the configuration file. You can also receive account status notifications when an event occurs during password policy processing, such as when accounts are locked out, accounts expire, passwords expire, and so on.

Oracle Unified Directory supports the following alert handlers:

- JMX alert handler for JMX notifications
- SMTP alert handler for email notifications.

The following topics describe how to manage the alert handler configuration:

- Managing Alert Handlers Using `dsconfig`
- Managing Alert Handlers Using OUDSM
- Supported Alert Types

35.4.1.1 Managing Alert Handlers Using `dsconfig`

You use the `dsconfig` command to manage the alert handler configuration. For information about configuring alerts by using the OUDSM interface, see Managing Alert Handlers Using OUDSM.

This section contains the following topics:

- Viewing the Configured Alert Handlers
- Enabling an Alert Handler
- Creating a New Alert Handler
- Deleting an Alert Handler
- Controlling the Allowed Alert Types

35.4.1.1.1 Viewing the Configured Alert Handlers

Oracle Unified Directory stores alert handlers information in the configuration file under the `cn=Alert Handlers,cn=config` subtree. You can access the information using the `dsconfig` command.

To display a list of alert handlers, run the following `dsconfig` command:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
 list-alert-handlers

Alert Handler     : Type : enabled
------------------:------:--------
JMX Alert Handler : jmx  : false
```

35.4.1.1.2 Enabling an Alert Handler

The JMX alert handler is disabled by default. Before you begin, you must configure JMX on the server. For more information, see Monitoring the Server Using JConsole.

1. To list the alert handler's properties, use the `dsconfig` command as follows.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-alert-handler-prop --handler-name "JMX Alert Handler"

Property            : Value(s)
--------------------:---------------------------------------------
disabled-alert-type : -
enabled             : false
enabled-alert-type  : -
```

2. To enable the alert handler, use `dsconfig` as follows.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-alert-handler-prop --handler-name "JMX Alert Handler" --set enabled:true
```

3. Verify the change by using `dsconfig`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  get-alert-handler-prop --handler-name "JMX Alert Handler"

Property            : Value(s)
--------------------:---------------------------------------------
disabled-alert-type : -
enabled             : true
enabled-alert-type  : -
```

35.4.1.1.3 Creating a New Alert Handler

The example in this section configures a new SMTP handler. Before starting this procedure, you must have configured an SMTP server for Oracle Unified Directory.

1. Specify an SMTP server by setting the smtp-server global configuration property. For more information, see Configuring Task Notification.
2. To create an alert handler run `dsconfig` with the `create-alert-handler` subcommand.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-alert-handler --handler-name "my SMTP Handler" --type smtp \
  --set enabled:true --set message-body:"Alert Type: %%alert-type%%
  \n\nAlert ID: %%alert-id%%\n\nAlert Message: %%alert-message%%" \
  --set message-subject:"Alert Message" \
  --set recipient-address:directorymanager@example.com \
  --set sender-address:OUD-Alerts@directory.example.com
```

3. View the list of alert handlers as follows.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-alert-handlers
```

35.4.1.1.4 Deleting an Alert Handler

You use the `dsconfig delete-alert-handler` command to delete an alert handler. The following example removes the JMX alert handler.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  delete-alert-handler --handler-name "JMX Alert Handler"
```

You can simply *disable* an alert handler instead of deleting it. In this case, the alert handler is available if you want to enable it again in the future. For more information, see Controlling the Allowed Alert Types.

35.4.1.1.5 Controlling the Allowed Alert Types

Oracle Unified Directory, by default supports alert types are allowed. If you specify a value for the `enabled-alert-type` property, only alerts with one of those types are allowed. If you specify a value for the `disabled-alert-type` property, all alert types except for the values in that property are allowed. Alert types are specified by their Java class, as shown in this example.

For a list of all supported alert types, see Supported Alert Types.

To disable an alert type, specify its Java class as a value of the `disabled-alert-type` property.

This command disables the startup alert from the JMX Alert Handler.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-alert-handler-prop --handler-name "JMX Alert Handler" \
  --set disabled-alert-type:org.opends.server.DirectoryServerStarted
```

35.4.1.2 Managing Alert Handlers Using OUDSM

You use OUDSM to manage the alert handler configuration. For information about configuring alert handlers by using `dsconfig`, see Managing Alert Handlers Using `dsconfig`.

This section contains the following topics:

- Creating an Alert Handler
- Modifying an Alert Handler
- Deleting an Alert Handler

35.4.1.2.1 Creating an Alert Handler

To create an alert handler using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. From the **Create** menu, select **Alert Handler**.
4. Select the type of alert handler that you want to create:
   - **JMX.** This alert handler is used to generate JMX notifications to alert administrators of significant events that occur within the server.

- **SMTP.** This alert handler is used to send e-mail messages to notify administrators of significant events that occur within the server.

5. Enter the properties to configure the connection handler in the right hand pane.

   The configurable properties will depend on the type of alert handler that you have selected. For a comprehensive list of all configurable properties, and their allowed values, see "The Alert Handler Configuration" in the *Configuration Reference for Oracle Unified Directory.*

> **Note:**
>
> By default, all alert types are allowed. If you specify one or more values in the **Enabled Alert Type** field, only alerts with one of those types are allowed. If you specify one or more values in the **Disabled Alert Type** field, all alert types except for the values in that field are allowed.
>
> For a list of all supported alert types, see Supported Alert Types.

6. When you have configured the required properties for your specific alert handler type, click **Create**.

35.4.1.2.2 Modifying an Alert Handler

To modify an existing alert handler using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Alert Handlers** element.
5. Select the alert handler whose properties you want to modify.
6. The properties are display in the right hand pane.
7. When you have modified the required properties, click **Apply**.

35.4.1.2.3 Deleting an Alert Handler

To delete an existing alert handler using OUDSM:

1. Connect to the directory server from OUDSM, as described in Connecting to the Server Using OUDSM.
2. Select the **Configuration** tab.
3. Expand the **General Configuration** element.
4. Expand the **Alert Handlers** element.
5. Select the alert handler that you want to delete and click the **Delete configuration** icon.
6. You are prompted to confirm the deletion. Click **Yes**.

35.4.1.3 Supported Alert Types

The server sends out message alerts when an alert type event occurs in the system. The supported alert types are defined in the following table.

Alert Type	Description
Access Control Disabled Java Class: `org.opends.server.AccessControlDisabled`	Notify administrator that the access control handler has been disabled.
Access Control Enabled Java Class: `org.opends.server.AccessControlEnabled`	Notify administrator that the access control handler has been enabled.
Access Control Parse Failed Java Class: `org.opends.server.authorization.dseecompat.ACIParseFailed`	Notify administrator if the Oracle Directory Server Enterprise Edition compatible access control subsystem failed to correctly parse one or more ACI rules when the server is first started.
Access Control Modified Java Class: `org.opends.server.authorization.dseecompat.AciModified`	Notify administrator if the Oracle Directory Server Enterprise Edition compatible access control subsystem detected that one or more ACI rules have been modified.
Backend Environment Unusable Java Class: `org.opends.server.BackendRunRecovery`	Notify administrator that the JE back end throws a `RunRecoveryException` and the directory server must be restarted.
Cannot Copy Schema Files Java Class: `org.opends.server.CannotCopySchemaFiles`	Notify administrator if a problem occurs while attempting to create copies of the existing schema configuration before making a schema update, and the schema configuration is left in a potentially inconsistent state.
Cannot Find Recurring Task Java Class: `org.opends.server.CannotFindRecurringTask`	Notify administrator if the directory server cannot locate a recurring task definition to schedule the next iteration once the previous iteration has completed.
Cannot Rename Current Task File Java Class: `org.opends.server.CannotRenameCurrentTaskFile`	Notify administrator if the directory server cannot rename the current tasks backing file in the process of trying to write an updated version.
Cannot Rename New Task File Java Class: `org.opends.server.CannotRenameNewTaskFile`	Notify administrator if the directory server cannot rename the new tasks backing file into place.
Cannot Schedule Recurring Iteration Java Class: `org.opends.server.CannotScheduleRecurringIteration`	Notify administrator if the directory server cannot schedule an iteration of a recurring task.
Cannot Write Configuration Java Class: `org.opends.server.CannotWriteConfig`	Notify administrator if the directory server cannot write its updated configuration for some reason and so the server cannot exhibit the new configuration if it is restarted.
Cannot Write New Schema Files Java Class: `org.opends.server.CannotWriteNewSchemaFiles`	Notify administrator if a problem occurs while attempting to write new versions of the server schema configuration files, and the schema configuration is left in a potentially inconsistent state.
Cannot Write Task File Java Class: `org.opends.server.CannotWriteTaskFile`	Notify administrator if the directory server cannot write an updated tasks backing file for some reason.

Alert Type	Description
Distribution Backend Does Not Support PreRead Control Java Class: `com.sun.dps.server.distribution.globalindex.UnsupportedDirectoryBackend`	Notify administrators if the distribution cannot maintain the content of the global index catalog. This will happen \ if one or more servers do not support the Pre-Read Entry Control (RFC 4527)
Entering Lockdown Mode Java Class: `org.opends.server.EnteringLockdownMode`	Notify administrator that the directory server is entering lockdown mode, in which only root users will be allowed to perform operations and only over the loopback address.
LDAP Connection Handler Consecutive Failures Java Class: `org.opends.server.LDAPHandlerDisabledByConsecutiveFailures`	Notify administrator of consecutive failures that have occurred in the LDAP connection handler that have caused it to become disabled.
LDAP Connection Handler Uncaught Error Java Class: `org.opends.server.LDAPHandlerUncaughtError`	Notify administrator of uncaught errors in the LDAP connection handler that have caused it to become disabled.
LDAP Server Extension Failed Java Class: `com.sun.dps.server.workflowelement.proxyldap.LDAPServerExtension.LDAPServerExtensionDown`	Notify administrator that the LDAP Server Extension has been detected as Down.
LDAP Server Extension is Up Java Class: `com.sun.dps.server.workflowelement.proxyldap.LDAPServerExtension.LDAPServerExtensionUp`	Notify administrator that the LDAP Server Extension has been detected as UP.
LDIF Backend Cannot Write Update Java Class: `org.opends.server.LDIFBackendCannotWriteUpdate`	Notify administrator that an LDIF back end was unable to store an updated copy of the LDIF file after processing a write operation.
LDIF ConnHandler Parse Error Java Class: `org.opends.server.LDIFConnectionHandlerParseError`	Notify administrator that the LDIF connection handler encountered an unrecoverable error while attempting to parse an LDIF file.
LDIF ConnHandler IO Error Java Class: `org.opends.server.LDIFConnectionHandlerIOError`	Notify administrator that the LDIF connection handler encountered an I/O error that prevented it from completing its processing.
Leaving Lockdown Mode Java Class: `org.opends.server.LeavingLockdownMode`	Notify administrator that the directory server is leaving lockdown mode.
Manual Config Edit Handled Java Class: `org.opends.server.ManualConfigEditHandled`	Notify administrator if the directory server detects that its configuration has been manually edited with the server online and those changes were overwritten by another change made through the server. The manually-edited configuration will be copied off to another location.

Alert Type	Description
Manual Config Edit Lost Java Class: `org.opends.server.ManualConfigEditLost`	Notify administrator if the directory server detects that its configuration has been manually edited with the server online and those changes were overwritten by another change made through the server. The manually-edited configuration could not be preserved due to an unexpected error.
New route elected by the SaturationLoadBalancingAlgorithm Java Class: `com.sun.dps.server.SaturationLoadBalancer`	Notify administrator that a new route has been elected as active route by the saturation load balancing algorithm.
New route elected by the FailoverLoadBalancingAlgorithm Java Class: `com.sun.dps.server.FailoverLoadBalancer`	Notify administrator that a new route has been elected as the active route by the failover load balancing algorithm.
Replication Unresolved Conflict Java Class: `org.opends.server.replication.UnresolvedConflict`	Notify administrator if the multimaster replication cannot automatically resolve a conflict.
Server Started Java Class: `org.opends.server.DirectoryServerStarted`	Notify administrator that the directory server has completed its startup process.
Server Shutdown Java Class: `org.opends.server.DirectoryServerShutdown`	Notify administrator that the directory server has begun the process of shutting down.
State change for a Saturation Load Balancing Route Java Class: `com.sun.dps.server.SaturationLoadBalancer`	Notify administrator that the saturation load balancing route state has changed (either from saturated to not saturated or from not saturated to saturated).
Uncaught Exception Java Class: `org.opends.server.UncaughtException`	Notify administrator if a directory server thread has encountered an uncaught exception that caused the thread to terminate abnormally. The impact that this problem has on the directory server depends on which thread was impacted and the nature of the exception.
Unique Attr Sync Conflict Java Class: `org.opends.server.UniqueAttributeSynchronizationConflict`	Notify administrator that a unique attribute conflict has been detected during synchronization processing.
Unique Attr Sync Error Java Class: `org.opends.server.UniqueAttributeSynchronizationError`	Notify administrator that an error occurred while attempting to perform unique attribute conflict detection during synchronization processing.
Unsupported Directory Backend Java Class: `com.sun.dps.server.distribution.globalindex.UnsupportedDirectoryBackend`	Notify administrator that the distribution cannot maintain the content of the global index catalog. This will happen if one or more servers do not support the Pre-Read Entry Control (RFC 4527).

35.4.2 Managing Account Status Notification Handlers

Account status notification handlers provide alerts on events during password policy processing. By default, the Error Log Account Status Notification handler is set to enabled upon initial configuration.

The server writes a message to the server error log when one of the following events has been configured in the password policy and occurs during password policy processing:

- account-temporarily-locked
- account-permanently-locked
- account-unlocked
- account-idle-locked
- account-reset-locked
- account-disabled
- account-expired
- password-expired
- password expiring
- password-reset
- password-changed

The error log is located at `instance-dir/OUD/logs/errors`.

This section contains the following topics:

- Viewing the Configured Account Status Notification Handlers
- Enabling Account Status Notification Handlers
- Creating a New Account Status Notification Handler
- Deleting an Account Status Notification Handler
- Customizing Message Template Files for SMTP Account Status Notification Handlers

35.4.2.1 Viewing the Configured Account Status Notification Handlers

You use the `list-account-status-notification-handlers` subcommand of `dsconfig` command to view the status of the notification handler.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-account-status-notification-handlers

Account Status Notification Handler : Type      : enabled
------------------------------------:-----------:--------
Error Log Handler                   : error-log : true
SMTP Handler                        : smtp      : false
```

35.4.2.2 Enabling Account Status Notification Handlers

You enable an existing account status notification handler using the `dsconfig` command. By default, the directory server enables the Error Log Handler when the server is initially configured. This example enables the SMTP notification handler.

1. To view the `enabled` property use `dsconfig` with the `get-account-status-notification-handler-prop` subcommand.

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     get-account-status-notification-handler-prop --handler-name "SMTP Handler" \
   ```

```
  --property enabled

Property : Value(s)
---------:---------
enabled  : false
```

2. To enable the notification handler use `dsconfig` with the `set-account-status-notification-handler-prop` subcommand.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  set-account-status-notification-handler-prop --handler-name "SMTP Handler" \
  --set property:enabled
```

35.4.2.3 Creating a New Account Status Notification Handler

To create an account status notification handler using the dsconfig command:

1. Use `dsconfig` with the `create-account-status-notification-handler` subcommand to create the handler.

   When you specify the type, you can use either `error-log` or `generic` (default).

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  create-account-status-notification-handler \
  --handler-name "My Password Reset Logger" --type error-log \
  --set enabled:true --set account-status-notification-type:password-reset
```

2. Use `dsconfig` to view the list of account status notification handlers.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  list-account-status-notification-handlers

Account Status Notification Handler : Type      : enabled
------------------------------------:-----------:--------
Error Log Handler                   : error-log : true
my Password Reset Logger            : error-log : true
SMTP Handler                        : smtp      : false
```

35.4.2.4 Deleting an Account Status Notification Handler

You can disable an account status notification handler instead of deleting it. In this case, the alert handler is available if you want to enable it again in the future.

You can remove an account status notification handler entirely by using `dsconfig`.

Use `dsconfig` with the `delete-account-status-notification-handler` subcommand.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
  delete-account-status-notification-handler \
  --handler-name "My Password Reset Logger"
```

35.4.2.5 Customizing Message Template Files for SMTP Account Status Notification Handlers

You can customize the message template files, which contain the message templates used to generate email notification messages.

Run the `dsconfig` command to view the available message template files for the SMTP account status notification handler, as follows:

```
$ dsconfig -h localhost -p 4444  -D "cn=Directory Manager" -j pwd-file --
trustALL \
get-account-status-notification-handler-prop --handler-name "SMTP Handler" --
property message-template-file
```

The following output appears:

```
Property                : Value(s)
------------------------:---------------------------------------------------
---------------
message-template-file   : account-disabled:config/messages/account-
disabled.template,
                        : account-enabled:config/messages/account-
enabled.template,
                        : account-expired:config/messages/account-
expired.template,
                        : account-idle-locked:config/messages/account-idle-
locked.template,
                        : account-permanently-locked:config/messages/account-
permanently-locked.template,
                        : account-reset-locked:config/messages/account-reset-
locked.template,
                        : account-temporarily-locked:config/messages/account-
temporarily-locked.template,
                        : account-unlocked:config/messages/account-
unlocked.template,
                        : password-changed:config/messages/password-
changed.template,
                        : password-expired:config/messages/password-
expired.template,
                        : password-expiring:config/messages/password-
expiring.template
                        : password-reset:config/messages/password-
reset.template
```

The `message-template-file` option can be configured using the preceding template files. The files are located at *`OUD_INSTANCE`*`/OUD/config/messages`. These template files already have the default email body text for all account activities.

Using placeholders, you can customize message templates to include dynamic parameters such as the user's DN or other attributes, notification type, and so on. The supported list of placeholders is provided as follows.

Placeholder	Description
%%notification-type%%	Specifies the type of the notification.
%%notification-message%%	Specifies the subject of the email.
%%notification-user-dn%%	Specifies the `dn` of the user.
%%notification-user-attr:sn%%	Specifies the `sn` of the user.
%%notification-user-attr:uid%%	Specifies the `uid` of the user.
%%notification-property:old-password%%	This is applicable for the password-changed notification type.
%%notification-property:new-password%%	This is applicable for the password-changed notification type.
%%notification-user-attr:<attribute_name>%%	Specifies any other valid <attribute_name> for the directory entry. For example, %%notification-user-attr:mail%%, %%notification-user-attr:phone%%, and so on.

35.5 Monitoring the Server with LDAP

Oracle Unified Directory provides a variety of methods to monitor the current state of the server for debugging or troubleshooting purposes.

The topics in this section assume that you have configured monitoring providers on the server. For more information, see Configuring Monitor Providers.

You can monitor the server over LDAP in several ways. These are described in the following sections:

- Viewing Monitoring Information Using the `cn=monitor` Entry
- Monitoring Using the `manage-tasks` Command
- Monitoring the Server Using JConsole
- Accessing Logs

35.5.1 Viewing Monitoring Information Using the `cn=monitor` Entry

The directory server records system, performance, and version information as an entry with the base DN of `cn=monitor`. This entry provides useful performance metrics and server state information that you can use to monitor and debug a directory server instance.

You can access the `cn=monitor` suffix over the administration port only. There are advantages to using the administration port to access monitoring information. The main advantage of the administration connector is the separation of user traffic and administration traffic.

For example, if you monitor the number of connections on the LDAP Connection Handler (`"cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port port-number,cn=monitor"`) over the regular LDAP port, your monitoring data are "polluted" by the monitoring request itself. All of the examples in this section use the administration port, over SSL. For more information, see Managing Administration Traffic to the Server.

This section includes the following topics:

- Overview of Monitored Attributes in the Proxy
- Viewing the Available Monitoring Information

- Monitoring General-Purpose Server Information
- Monitoring System Information
- Monitoring Version Information
- Monitoring the User Root Back End
- Monitoring the Backup Back End
- Monitoring the Tasks Back End
- Monitoring the `monitor` Back End
- Monitoring the `adminRoot` Back End
- Monitoring the `ads-truststore` Back End
- Monitoring Client Connections
- Monitoring the LDAP Connection Handler
- Monitoring LDAP Connection Handler Statistics
- Monitoring Connections on the LDAP Connection Handler
- Monitoring the Administration Connector
- Monitoring Administration Connector Statistics
- Monitoring Connections on the Administration Connector
- Monitoring the LDIF Connection Handler
- Monitoring the Work Queue
- Monitoring JVM Stack Trace Information
- Monitoring the JVM Memory Usage
- Monitoring the `userRoot` Database Environment
- Managing the Database Cache
- Monitoring the Entry Cache
- Monitoring Network Groups
- Monitoring Distribution
- Monitoring Load Balancing
- Monitoring Remote LDAP Servers
- Monitoring a Global Index
- Monitoring a Global Index Catalog

35.5.1.1 Overview of Monitored Attributes in the Proxy

Monitoring information related to the proxy can be collected at the level under `cn=Monitor` for dozens of attributes, including those relating to the following:

- Workflows: `cn=workflow,cn=monitor`
- Network Groups: `cn=Network Groups,cn=monitor`
- Load balancers: `cn=load balancing,cn=monitor`
- Distributions: `cn=distribution,cn=monitor`
- Global Index Catalogs: `cn=Global Index Catalogs,cn=monitor`

- Client Connections: `cn=Client Connections,cn=monitor` or under `cn=Client Connections,cn=LDAP Connection Handler` *0.0.0.0* `port` *port number*`,cn=monitor`
- LDAP Connection Handler: `cn=LDAP Connection Handler` *0.0.0.0* `port` *port number*`,cn=monitor`
- LDAP Connection Handler Statistics: `cn=LDAP Connection Handler` *0.0.0.0* `port`*port number* `statistics,cn=monitor`
- SNMP Connection Handler: `cn=SNMP Connection Handler,cn=Monitor`
- JMX Connection Handler: `cn=JMX Connection Handler` *port number*`,cn=monitor`
- Administration Connector: `cn=Administration Connector` *0.0.0.0* `port` *port number*`,cn=monitor`
- System Information: `cn=System Information,cn=monitor`
- Version: `cn=Version,cn=monitor`
- Back-end LDAP servers: `cn=LDAP Servers,cn=monitor`
- JVM stack traces: `cn=JVM Stack Trace,cn=monitor`
- JVM memory usage: `cn=JVM Memory Usage,cn=Monitor`
- SNMP: `cn=SNMP,cn=Monitor`
- Backend Backup: `cn=backup Backend,cn=monitor`
- Monitoring of back-end data: `cn=monitor Backend,cn=monitor`
- Tasks on the Backend Backup: `cn=backup Backend,cn=monitor`
- Entry caches: `cn=Entry Caches,cn=monitor`
- Work queues: `cn=Work Queue,cn=monitor`

Other attributes are monitored under each of the above in the dn tree. For example, client connections are monitored under both `cn=Client Connections,` *0.0.0.0* `port` *port number*`,cn=monitor` and under `cn=Client Connections,cn=Administration Connector` *0.0.0.0* `port` *port number*`, cn=monitor`.

A workflow element is monitored under the part of the tree to which that workflow element relates. For example, a load balancing workflow element can be monitored as `cn=load-bal-route1,cn=load balancing,cn=monitor`

Hundreds of statistics are collected by the proxy for monitoring. For example, for the persistent search function, `psearchCount` lists the number of persistent search operations and `psearchTotalCount` lists the number of persistent search operations since the last server restart.

You can list all of these statistics by using the `ldapsearch` command on the `cn=monitor` entry, as described in Viewing the Available Monitoring Information. Access to the `cn=monitor` entry is restricted to users who have the bypass ACI privilege.

The following procedures use the `ldapsearch` command at the command line interface.

To view status information on the replication of global indexes, you can use the `gicadm status-replication` command. For more information, see Viewing the Status of a Replicated Global Index Catalog Configuration

35.5.1.2 Viewing the Available Monitoring Information

Use the `ldapsearch` command to inspect the attributes of `cn=monitor`. This example lists the base DNs of each monitor entry.

Run the `ldapsearch` command with a search scope of `sub` and the search attribute `1.1`.

This search attribute indicates that no attributes should be included in the matching entries.

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s sub -b "cn=monitor" "(objectclass=*)" "1.1"
dn: cn=monitor
dn: cn=Client Connections,cn=monitor
dn: cn=ads-truststore Backend,cn=monitor
dn: cn=Network Groups,cn=monitor
dn: cn=internal,cn=Network Groups,cn=monitor
dn: cn=default,cn=Network Groups,cn=monitor
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor
dn: cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
dn: cn=Client Connections,cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
dn: cn=backup Backend,cn=monitor
dn: cn=Version,cn=monitor
dn: cn=Work Queue,cn=monitor
dn: cn=System Information,cn=monitor
dn: cn=userRoot Database Environment,cn=monitor
dn: cn=tasks Backend,cn=monitor
dn: cn=adminRoot Backend,cn=monitor
dn: cn=userRoot Backend,cn=monitor
dn: cn=schema Backend,cn=monitor
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
dn: cn=admin,cn=Network Groups,cn=monitor
dn: cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
dn: cn=JVM Memory Usage,cn=monitor
dn: cn=Administration Connector 0.0.0.0 port 4444 Statistics,cn=monitor
dn: cn=JVM Stack Trace,cn=monitor
dn: cn=Entry Caches,cn=monitor
dn: cn=monitor Backend,cn=monitor
```

35.5.1.3 Monitoring General-Purpose Server Information

Use the `ldapsearch` command with a base DN of `"cn=monitor"` to monitor general-purpose information for a server.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=monitor" "(objectclass=*)"
```

Output will be similar to the following:

```
dn: cn=monitor
currentTime: 20240111082026Z
totalConnections: 7
vendorName: Oracle Corporation
cn: monitor
vendorVersion: Oracle Unified Directory 14.1.2.1.241104
version: Oracle Unified Directory 14.1.2.1.241104
currentConnections: 1
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
```

```
upTime: 57 days 21 hours 18 minutes 30 seconds
startTime: 20240110110156Z
maxConnections: 1
productName: Oracle Unified Directory
```

35.5.1.4 Monitoring System Information

You use the ldapsearch command to monitor the system information.

Run the `ldapsearch` command with the base DN `"cn=System Information,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=System Information,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=System Information,cn=monitor
instancePath: /export/home/oracle/OUD/asinst_1/OUD
javaVersion: 1.8.0_211-b12
jvmArchitecture: 64-bit
jvmArguments: "-Dorg.opends.server.scriptName=start-ds"
jvmVersion: 24.65-b04
classPath: /export/home/oracle/OUD/asinst_1/OUD/classes:/export/home/oracle/OUD/
OracleUnifiedDirectory/winlib/classpath.jar:/export/home/oracle/OUD/asinst_1/OU
 D/lib/*.jar
usedMemory: 69402624
freeUsedMemory: 23084640
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
javaVendor: Oracle Corporation
operatingSystem: Linux 2.6.32-200.13.1.el5uek amd64
cn: System Information
systemName: sboy
installPath: /export/home/oracle/OUD/OracleUnifiedDirectory
workingDirectory: /export/home/oracle/OUD/asinst_1/OUD/bin
availableCPUs: 2
maxMemory: 922746880
javaHome: /usr/lib/jvm/jdk8/jre
jvmVendor: Oracle Corporation
```

35.5.1.5 Monitoring Version Information

Use the `ldapsearch` command to monitor the version information.

Run the `ldapsearch` command with base DN `"cn=Version,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -b "cn=Version,cn=Monitor" "(objectclass=*)"
```

The output will be similar to the following:

```
dn: cn=version,cn=monitor
componentVersion: 4
fullVersion: Oracle Unified Directory 14.1.2.1.241104
buildID: 20240215065722Z
shortName: OUD
compactVersion: OUD-14.1.2.1.241104
cn: Version
version: Oracle Unified Directory 14.1.2.1.241104
objectClass: top
```

```
objectClass: ds-monitor-entry
objectClass: extensibleObject
labelIdentifier: 2402130001
maintenanceVersion: 2
majorVersion: 12
releaseVersion: 1
platformVersion: 240213
productName: Oracle Unified Directory
```

35.5.1.6 Monitoring the User Root Back End

The `userRoot` back end is the back-end database (the JE environment) for your data. The monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Run the `ldapsearch` command with base DN `"cn=userRoot Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
--useSSL --trustAll -s base -b "cn=userRoot Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=userRoot Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: FALSE
cn: userRoot Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 2002
ds-backend-id: userRoot
ds-base-dn-entry-count: 2002 dc=example,dc=com
ds-backend-base-dn: dc=example,dc=com
```

35.5.1.7 Monitoring the Backup Back End

You use the ldapsearch command to monitor the backup back end.

Run the `ldapsearch` command with base DN `"cn=backup Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file
--useSSL --trustAll -s base -b "cn=backup Backend,cn=monitor" "(objectclass=*)" \
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=backup Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: backup Backend
ds-backend-writability-mode: disabled
ds-backend-entry-count: 1
ds-backend-id: backup
ds-base-dn-entry-count: 1 cn=backups
ds-backend-base-dn: cn=backups
```

35.5.1.8 Monitoring the Tasks Back End

Tasks are administrative functions (such as `import-ldif`, `export-ldif`, `backup`, and `restore`) that can be scheduled for processing at some future date or on a recurring basis. The monitor

displays the tasks back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Run the `ldapsearch` command with base DN `"cn=Tasks Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=Tasks Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=tasks Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: tasks Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 3
ds-backend-id: tasks
ds-base-dn-entry-count: 3 cn=tasks
ds-backend-base-dn: cn=tasks
```

35.5.1.9 Monitoring the `monitor` Back End

The monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Run the `ldapsearch` command with base DN `"cn=monitor Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=monitor Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=monitor Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: monitor Backend
ds-backend-writability-mode: disabled
ds-backend-entry-count: 25
ds-backend-id: monitor
ds-base-dn-entry-count: 25 cn=monitor
ds-backend-base-dn: cn=monitor
```

35.5.1.10 Monitoring the Schema Back End

This monitor displays the schema back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Run the `ldapsearch` command with base DN `"cn=schema Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=schema Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=schema Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
```

```
ds-backend-is-private: TRUE
cn: schema Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 1
ds-backend-id: schema
ds-base-dn-entry-count: 1 cn=schema
ds-backend-base-dn: cn=schema
```

35.5.1.11 Monitoring the `adminRoot` Back End

This monitor displays the `adminRoot` back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Use the `ldapsearch` command with base DN `"cn=adminRoot Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=adminRoot Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=adminRoot Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: adminRoot Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 7
ds-backend-id: adminRoot
ds-base-dn-entry-count: 7 cn=admin data
ds-backend-base-dn: cn=admin data
```

35.5.1.12 Monitoring the `ads-truststore` Back End

The `ads-truststore` holds a mirror, or copy, of the remote Administrative Directory Service (ADS) host's ADS key entry, so that the new instance can establish trust with existing servers in the ADS domain. The monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

Run the `ldapsearch` command with base DN `"cn=ads-truststore Backend,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=ads-truststore Backend,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=ads-truststore Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: ads-truststore Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 3
ds-backend-id: ads-truststore
ds-base-dn-entry-count: 3 cn=ads-truststore
ds-backend-base-dn: cn=ads-truststore
```

35.5.1.13 Monitoring Client Connections

This monitor represents *all* of the open client connections. Its contents are different to those of the DN `"cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor"`, which describes the open client connections on the LDAP connection handler only.

Run the `ldapsearch` command with base DN `"cn=Client Connections,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=Client Connections,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=Client Connections,cn=monitor
connection: connID="11" connectTime="20090702125632Z" source="198.51.100.0:54044"
destination="198.51.100.23:1389" ldapVersion="3" authDN="cn=Directory Manager,cn=Root
DNs,
cn=config" security="none" opsInProgress="1"
cn: Client Connections
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
```

35.5.1.14 Monitoring the LDAP Connection Handler

The LDAP connection handler is used to interact with clients over LDAP.

Run the `ldapsearch` command with base DN `"cn=LDAP Connection Handler 0.0.0.0 port port-number,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base \
  -b "cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor" \
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
ds-connectionhandler-listener: 0.0.0.0:1389
ds-connectionhandler-num-connections: 1
ds-connectionhandler-protocol: LDAP
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
ds-mon-config-dn: cn=ldap connection handler,cn=connection handlers,cn=config
cn: LDAP Connection Handler 0.0.0.0 port 1389
ds-connectionhandler-connection: connID="22" connectTime="20120302133936Z"
source="198.51.100.0:39574" destination="198.51.100.23:1389" ldapVersion="3"
authDN="cn=Directory Manager,cn=Root DNs,cn=config" security="none" opsInProgress="1"
```

35.5.1.15 Monitoring LDAP Connection Handler Statistics

You use the ldapsearch command to monitor LDAP connection handler statistics.

Run the `ldapsearch` command with base DN `"cn=LDAP Connection Handler 0.0.0.0 port port-number Statistics,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base \
  -b "cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor" \
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor
objectClass: ds-monitor-entry
objectClass: top
objectClass: extensibleObject
operationsCompleted: 37
compareRequests: 0
bytesWritten: 99488
extendedRequests: 0
addRequests: 0
bindRequests: 19
...(more output)
```

35.5.1.16 Monitoring Connections on the LDAP Connection Handler

This monitor represents the open client connections on the LDAP connection handler.

Run the `ldapsearch` command with base DN `"cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port port-number,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
--useSSL --trustAll \
-b "cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389 \
cn=monitor" \
"(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
connection: connID="0" connectTime="20090706084747Z" source="198.51.100.0:57523"
 destination="198.51.100.0:1389" ldapVersion="3" authDN="" security="none"
 opsInProgress="0"
connection: connID="1" connectTime="20090706084747Z" source="198.51.100.0:57524"
 destination="198.51.100.0:1389" ldapVersion="3" authDN="" security="none"
 opsInProgress="0"
connection: connID="2" connectTime="20090706084747Z" source="198.51.100.0:57525"
 destination="198.51.100.0:1389" ldapVersion="3" authDN="" security="none"
 opsInProgress="0"
connection: connID="3" connectTime="20090706084747Z" source="198.51.100.0:57526"
 destination="198.51.100.0:1389" ldapVersion="3" authDN="" security="none"
 opsInProgress="0"
connection: connID="4" connectTime="20090706084747Z" source="198.51.100.0:57527"
 destination="198.51.100.0:1389" ldapVersion="3" authDN="" security="none"
 opsInProgress="0"
```

35.5.1.17 Monitoring the Administration Connector

This monitor provides basic information about the administration connector. For more information, see Managing Administration Traffic to the Server.

Run the `ldapsearch` command with base DN `"cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -b "cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor" \
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
dn: cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor
ds-connectionhandler-listener: 0.0.0.0:4444
ds-connectionhandler-num-connections: 0
ds-connectionhandler-protocol: LDAPS
cn: Administration Connector 0.0.0.0 port 4444
ds-mon-config-dn: cn=administration connector,cn=config
```

35.5.1.18 Monitoring Administration Connector Statistics

This monitor provides extensive statistical information about operations that are performed through the administration connector. For more information, see Managing Administration Traffic to the Server.

Run the `ldapsearch` command with base DN `"cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll \
  -b "cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor" \
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=LDAP Administration Connector 0.0.0.0 port 4444 Statistics,cn=monitor
compareResponses: 0
connectionsClosed: 1
searchResultsDone: 4
ds-mon-resident-time-mod-operations-total-time: 92257568
extendedResponses: 0
bindRequests: 2
operationsAbandoned: 0
bytesWritten: 45056
addResponses: 0
addRequests: 0
ds-mon-resident-time-moddn-operations-total-time: 0
ds-mon-extended-operations-total-count: 0
ds-mon-moddn-operations-total-count: 0
modifyResponses: 1
operationsCompleted: 7
...(more output)...
```

35.5.1.19 Monitoring Connections on the Administration Connector

This monitor represents the open client connections on the Administration Connector.

Run the `ldapsearch` command with base DN `"cn=Client Connections,cn=LDAP Administration Connector 0.0.0.0 port port-number,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
--useSSL --trustAll \
-b "cn=Client Connections,cn=LDAP Administration Connector 0.0.0.0 \
```

```
port 4444,cn=monitor" \
"(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
dn: cn=Client Connections,cn=LDAP Administration Connector 0.0.0.0 port 4444,cn=monitor
connection: connID="339" connectTime="20120307075218Z" source="198.51.100.0:48213"
destination="198.51.100.0:4444" ldapVersion="3" authDN="" security="TLS"
 opsInProgress="1"
cn: Client Connections
```

35.5.1.20 Monitoring the LDIF Connection Handler

The LDIF connection handler is used to process changes that are read from an LDIF file, using internal operations. Monitoring information for the LDIF connection handler is only available if the connection handler is enabled.

Run the `ldapsearch` command with base DN `"cn=LDIF Connection Handler,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=LDIF Connection Handler,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
dn: cn=LDIF Connection Handler,cn=monitor
ds-connectionhandler-num-connections: 0
ds-connectionhandler-protocol: LDIF
ds-mon-config-dn: cn=ldif connection handler,cn=connection handlers,cn=config
cn: LDIF Connection Handler
```

35.5.1.21 Monitoring the Work Queue

The work queue keeps track of outstanding client requests and ensures that they are processed.

Run the `ldapsearch` command with base DN `"cn=Work Queue,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=Work Queue,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=Work Queue,cn=monitor
currentRequestBacklog: 0
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
requestsSubmitted: 25
cn: Work Queue
maxRequestBacklog: 0
averageRequestBacklog: 0
requestsRejectedDueToQueueFull: 0
```

35.5.1.22 Monitoring JVM Stack Trace Information

You can access JVM Stack Trace information for your directory server instance. This resource monitor is implemented in the `org.opends.server.monitors.StackTraceMonitorProvider` class and requires no custom configuration.

Run the `ldapsearch` command with the base DN `"cn=JVM Stack Trace,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=JVM Stack Trace,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, the beginning of the output will be similar to the following:

```
dn: cn=JVM Stack Trace,cn=monitor
cn: JVM Stack Trace
jvmThread: id=2 ---------- Reference Handler ----------
jvmThread: id=2 frame[0]=java.lang.Object.wait(Object.java:native)
jvmThread: id=2 frame[1]=java.lang.Object.wait(Object.java:485)
jvmThread: id=2 frame[2]=java.lang.ref.Reference$ReferenceHandler.run(Reference.
java:116)
jvmThread: id=3 ---------- Finalizer ----------
jvmThread: id=3 frame[0]=java.lang.Object.wait(Object.java:native)
jvmThread: id=3 frame[1]=java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java
:116)
jvmThread: id=3 frame[2]=java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java
:132)
jvmThread: id=3 frame[3]=java.lang.ref.Finalizer$FinalizerThread.run(Finalizer.j
ava:159)
jvmThread: id=4 ---------- Signal Dispatcher ----------
jvmThread: id=10 ---------- Time Thread ----------
jvmThread: id=10 frame[0]=sun.misc.Unsafe.park(Unsafe.java:native)
jvmThread: id=10 frame[1]=java.util.concurrent.locks.LockSupport.parkNanos(LockS
upport.java:198)
...(more output)...
```

35.5.1.23 Monitoring the JVM Memory Usage

You use the `ldapsearch` command to configure the JVM memory.

Run the `ldapsearch` command with base DN `"cn=JVM Memory Usage,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=JVM Memory Usage,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=JVM Memory Usage,cn=monitor
ps-eden-space-bytes-used-after-last-collection: 0
ps-mark-sweep-total-collection-count: 0
code-cache-bytes-used-after-last-collection: 0
ps-old-gen-current-bytes-used: 25260472
ps-perm-gen-bytes-used-after-last-collection: 0
ps-scavenge-recent-collection-duration: 3
ps-scavenge-total-collection-count: 17
ps-eden-space-current-bytes-used: 32001992
ps-perm-gen-current-bytes-used: 21179960
ps-old-gen-bytes-used-after-last-collection: 0
ps-mark-sweep-total-collection-duration: 0
ps-mark-sweep-average-collection-duration: 0
```

```
ps-scavenge-average-collection-duration: 26
ps-scavenge-total-collection-duration: 443
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
ps-mark-sweep-recent-collection-duration: 0
ps-survivor-space-bytes-used-after-last-collection: 622592
cn: JVM Memory Usage
code-cache-current-bytes-used: 2143680
ps-survivor-space-current-bytes-used: 622592
```

35.5.1.24 Monitoring the `userRoot` Database Environment

The `userRoot` database environment utilizes the Berkeley DB Java Edition back end. JE monitoring data (data under `cn=*Database Environment,cn=monitor`) is reliable only in the short term. During high server activity (for example, anywhere from an hour to several days depending on the counter), this data can overflow. In such cases, the JE monitoring data can reflect negative values or positive but incorrect values. This is a known issue and is expected to be fixed in the next major release of the Berkeley DB Java Edition. Oracle SR numbers 15979 and 15985 correspond to this issue.

Run the `ldapsearch` command with base DN `"cn=userRoot Database Environment,cn=monitor"` as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=userRoot Database Environment,cn=monitor" \
  "(objectclass=*)"
dn: cn=userRoot Database Environment,cn=monitor
```

Depending on your configuration, output will be similar to the following:

```
EnvironmentNTempBufferWrites: 0
EnvironmentNNodesExplicitlyEvicted: 0
EnvironmentCleanerBacklog: 0
EnvironmentTotalLogSize: 5386067
EnvironmentLockBytes: 2000
EnvironmentNFullBINFlush: 2
EnvironmentNBINsStripped: 0
EnvironmentLastCheckpointEnd: 5385359
TransactionNCommits: 24
EnvironmentNCleanerEntriesRead: 0
EnvironmentNRepeatFaultReads: 2
TransactionNXACommits: 0
EnvironmentNClusterLNsProcessed: 0
TransactionNBegins: 24
LockNOwners: 25
...(more output)...
```

35.5.1.25 Managing the Database Cache

The following topics describe the database cache and how to monitor the same:

- Overview of Database Cache
- Monitoring the Database Cache

35.5.1.25.1 Overview of Database Cache

The database (DB) cache is used to store Java Edition nodes. The DB cache is the critical component of your directory server's overall performance. Ensure that you tune and monitor the DB cache carefully. The DB cache includes the following nodes:

- Upper node
- Inner node
- Leaf node

The upper and inner nodes represents the internal B-tree structure and the leaf node represent the user entries. For best possible performance, it is recommended to have all the DB cache nodes in the DB cache. It is recommend to size the dbcache such that it contains at minimum the B-tree internal structure (the upper and inner nodes). If the dbcache is too short this can result in having lots of misses and frequent evictions which will badly affect directory server performance.

Tuning the size of the cache is done by:

- Setting the `dbcache-percent`
- Sizing appropriately the Oracle Unified Directory JVM heap and especially the old generation.

The following DB cache hits and miss counters are described below:

Counters	Description
`EnvironmentNUpperINsFetch`	Accumulated number of upper inner nodes fetched from the cache.
`EnvironmentNUpperINsFetchMiss`	Accumulated number of upper inner nodes miss.
`EnvironmentNBINsFetch`	Accumulated number of bottom inner nodes fetched from the cache.
`EnvironmentNBINsFetchMiss`	Accumulated number of upper inner nodes miss.
`EnvironmentNLNsFetch`	Accumulated number of leaf nodes fetched from the cache.
`EnvironmentNLNsFetchMiss`	Accumulated number of leaf nodes miss.

For Oracle Unified Directory to perform well, Oracle recommends having all the nodes in the dbcache, or at least having all the inner nodes in the dbcache.

As the values in cn=monitor are accumulations, it is important to compute deltas at regular interval (1mn for instance) and monitor the evolution of deltas over time. You must update the following:

```
DeltaNUpperINsMiss=EnvironmentNUpperINsFetchMiss - EnvironmentNUpperINsFetchMissPrev
DeltaNUpperINsFetch=EnvironmentNUpperINsFetch - EnvironmentNUpperINsFetchPrev
DeltaBINsMiss=EnvironmentNBINsFetchMiss - EnvironmentNBINsFetchMissPrev
DeltaBINsFetch=EnvironmentNBINsFetch - EnvironmentNBINsFetchPrev
DeltaNLNsMiss=EnvironmentNLNsFetchMiss - EnvironmentNLNsFetchMissPrev
DeltaNLNsFetch=EnvironmentNLNsFetch - EnvironmentNLNsFetchPrev
```

You can run the Oracle Unified Directory with a minimal level of performance. It is recommend to have the B-Tree structure in the dbcache, as described below:

```
((DeltaNUpperINsMiss/DeltaNUpperINsFetch)*100) as close to 0 as possible
((DeltaNBINsMiss/DeltaNBINsFetch)*100) as close to 0 as possible (< 5% remains
acceptable)
```

To have the best possible performance, Oracle recommends that Oracle Unified Directory also have user entries in the dbcache, i-e:

`((DeltaNLNsMiss/DeltaNLNsFetch)*100)` as close to `0` as possible.

Start with Deltas ratio close to `0` after the import is complete (and data primed) and with time the Deltas ratio grows due to the database growth (because of replication metadata, clean-min-utilizat° impact, growth of the entry (new apps) as well as the nb of entries). Consequently, it is recommended that you monitor the dbcache (by using custom scripts or UI) and take appropriate actions such as increasing the dbcache-percent or the Oracle Unified Directory JVM heap.

35.5.1.25.2 Monitoring the Database Cache

You use the `ldapsearch` command with base DN `cn=userRoot Database Environment,cn=monitor` to monitor the DB cache.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=userRoot Database Environment,cn=monitor" \
  "(objectclass=*)"
dn: cn=userRoot Database Environment,cn=monitor
```

Depending on your configuration, output will be similar to the following:

```
EnvironmentNTempBufferWrites: 0
EnvironmentNNodesExplicitlyEvicted: 0
EnvironmentCleanerBacklog: 0
EnvironmentTotalLogSize: 5386067
EnvironmentLockBytes: 2000
EnvironmentNFullBINFlush: 2
EnvironmentNBINsStripped: 0
EnvironmentLastCheckpointEnd: 5385359
TransactionNCommits: 24
EnvironmentNCleanerEntriesRead: 0
EnvironmentNRepeatFaultReads: 2
TransactionNXACommits: 0
EnvironmentNClusterLNsProcessed: 0
TransactionNBegins: 24
LockNOwners: 25
...(more output)...
```

35.5.1.26 Monitoring the Entry Cache

You can access the aggregated state of all active entry caches for your directory server instance by accessing the `cn=Entry Caches,cn=Monitor` entry. The server can also request the "per cache" monitor data for a given instance if the entry cache instances are enabled in the directory server configuration:

- `cn=FIFO Entry Cache,cn=Monitor`
- `cn=Soft Reference Entry Cache,cn=Monitor`
- `cn=File System Entry Cache,cn=Monitor`

Additionally, any arbitrarily named active entry cache instance should provide a monitor, which can be accessed by that instance name, for example `cn=Any Arbitrary Name Entry Cache,cn=Monitor`.

Use the `ldapsearch` command with base DN `"cn=Entry Caches,cn=monitor"`.

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file --useSSL \
  --trustAll -s base -b "cn=Entry Caches,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=Entry Caches,cn=monitor
entryCacheHits: 0
entryCacheTries: 0
currentEntryCacheCount: 0
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
entryCacheHitRatio: 0
cn: Entry Caches
...
```

35.5.1.27 Monitoring Network Groups

You use the `ldapsearch` command with the base DN `"cn=Network Groups,cn=monitor"` to monitor network groups.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
--useSSL --trustAll -b "cn=Network Groups,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-mon-branch
dn: cn=Network Groups,cn=monitor
dn: cn=admin,cn=Network Groups,cn=monitor
ds-mon-compare-operations-total-count: 0
ds-mon-failed-referrals-total-count: 15
ds-mon-unbind-operations-total-count: 13
ds-mon-followed-referrals-total-count: 34
ds-mon-violations-schema-total-count: Not implemented
ds-mon-bind-operations-total-count: 98
ds-mon-persistent-searchs-count: Not implemented
ds-mon-add-operations-total-count: 37
ds-mon-abandon-operations-total-count: 0
ds-mon-moddn-operations-total-count: 0
ds-mon-extended-operations-total-count: 0
ds-mon-searchsubtree-operations-total-count: 310
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
ds-mon-discarded-referrals-total-count: Not implemented
ds-mon-mod-operations-total-count: 1
ds-mon-forwarded-referrals-total-count: Not implemented
cn: admin
ds-mon-searchonelevel-operations-total-count: 92966
ds-mon-delete-operations-total-count: 0
```

```
dn: cn=default,cn=Network Groups,cn=monitor
...
```

35.5.1.28 Monitoring Distribution

You can use the `ldapsearch` command with the base DN `"cn=Distribution,cn=monitor"` to monitor distribution.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -j pwd-file \
  --useSSL --trustAll -b "cn=Distribution,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-mon-branch
dn: cn=distribution,cn=monitor

cn: distrib-we
ds-mon-searchonelevel-operations-total-count: 0
ds-mon-residenttime-bind-operations-max-time: 0
...

ds-mon-delete-operations-total-count: 0

dn: cn=algorithm,cn=distrib-we,cn=distribution,cn=monitor
ds-mon-residenttime-total-time: 0
ds-mon-residenttime-max-time: 0
cn: algorithm
ds-mon-runs-total-count: 0
ds-mon-residenttime-min-time: 0
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject

dn: cn=partitions,cn=algorithm,cn=distrib-we,cn=distribution,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-mon-branch

dn: cn=distrib-part1,cn=partitions,cn=algorithm,cn=distrib-we,cn=distribution,cn
 =monitor
...
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
ds-mon-modify-operations-total-count: 0
cn: distrib-part1
ds-mon-searchonelevel-operations-total-count: 0
ds-mon-delete-operations-total-count: 0

dn: cn=distrib-part2,cn=partitions,cn=algorithm,cn=distrib-we,cn=distribution,cn
 =monitor

...
```

35.5.1.29 Monitoring Load Balancing

You use the `ldapsearch` command with the base DN `"cn=load balancing,cn=monitor"` to monitor load balancing.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file \
  --useSSL --trustAll -b "cn=load balancing,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-mon-branch
dn: cn=load balancing,cn=monitor
dn: cn=load-bal-we1,cn=load balancing,cn=monitor
ds-mon-aborted-add-operations-total-count: 0
...
dn: cn=algorithm,cn=load-bal-we1,cn=load balancing,cn=monitor

dn: cn=routes,cn=algorithm,cn=load-bal-we1,cn=load balancing,cn=monitor
...
dn: cn=load-bal-route1,cn=routes,cn=algorithm,cn=load-bal-we1,cn=load
balancing,cn=monitor
...
dn: cn=load-bal-we2,cn=load balancing,cn=monitor
...
dn: cn=algorithm,cn=load-bal-we2,cn=load balancing,cn=monitor
...
dn: cn=routes,cn=algorithm,cn=load-bal-we2,cn=load balancing,cn=monitor

dn: cn=load-bal-route1,cn=routes,cn=algorithm,cn=load-bal-we2,cn=load
balancing,cn=monitor
...
cn: load-bal-route1

dn: cn=load-bal-route2,cn=routes,cn=algorithm,cn=load-bal-we1,cn=load
balancing,cn=monitor
...
cn: load-bal-route2

dn: cn=load-bal-route2,cn=routes,cn=algorithm,cn=load-bal-we2,cn=load
balancing,cn=monitor

cn: load-bal-route2
ds-mon-searchonelevel-operations-total-count: 9
ds-mon-delete-operations-total-count: 0
```

35.5.1.30 Monitoring Remote LDAP Servers

You use the `ldapsearch` command with the base DN `"cn=LDAP Servers,cn=monitor"` to monitor remote LDAP servers.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file \
  --useSSL --trustAll -b "cn=LDAP Servers,cn=monitor" "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-mon-branch
dn: cn=LDAP Servers,cn=monitor

dn: cn=proxy1,cn=LDAP Servers,cn=monitor
ds-mon-aborted-add-operations-total-count: 0
...
cn: proxy1
ds-mon-searchonelevel-operations-total-count: 0
...
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject

dn: cn=proxy2,cn=LDAP Servers,cn=monitor
ds-mon-aborted-add-operations-total-count: 0
...
cn: proxy2
ds-mon-searchonelevel-operations-total-count: 0
...
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
...
dn: cn=proxy3,cn=LDAP Servers,cn=monitor
...
cn: proxy3
ds-mon-searchonelevel-operations-total-count: 0
...
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
...
dn: cn=proxy4,cn=LDAP Servers,cn=monitor
...
cn: proxy4
...
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
```

35.5.1.31 Monitoring a Global Index

You use the `ldapsearch` command with the base DN `"cn=givenname,cn=gi-catalog,cn=Global Index Catalogs,cn=monitor"` to monitor the global index.

Ensure that `givenname` corresponds to the name of the indexed attribute (for example `cn`, if you indexed `cn`), and that `gi-catalog` corresponds to the name of the global index catalog.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file --useSSL \
  --trustAll -b "cn=givenname,cn=gi-catalog,cn=Global Index Catalogs,cn=monitor"
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=givenname,cn=gi-catalog,cn=Global Index Catalogs,cn=monitor
ds-mon-add-operations-min-time: 0
ds-mon-add-operations-aborted-count: 0
ds-mon-lookup-operations-min-time: 0
```

```
ds-mon-getpartitions-operations-total-count: 0
ds-mon-add-operations-max-time: 0
ds-mon-lookup-operations-total-count: 0
ds-mon-memorized-remove-operations-count: 0
ds-mon-remove-operations-aborted-count: 0
ds-mon-add-operations-total-time: 0
ds-mon-getpartitions-operations-aborted-count: 0
ds-mon-lookup-operations-total-time: 0
ds-mon-index-entries: 0
ds-mon-remove-operations-failed-count: 0
ds-mon-getpartitions-operations-min-time: 0
ds-mon-lookup-operations-max-time: 0
ds-mon-getpartitions-operations-average-time: 0
ds-mon-index-creation-date: 1252483187019
ds-mon-getpartitions-operations-last-access-date: 0
ds-mon-remove-operations-total-count: 0
ds-mon-lookup-operations-failed-count: 0
ds-mon-add-operations-failed-count: 0
ds-mon-remove-operations-min-time: 0
ds-mon-add-operations-average-time: 0
ds-mon-lookup-operations-aborted-count: 0
ds-mon-getpartitions-operations-total-time: 0
ds-mon-remove-operations-max-time: 0
ds-mon-getpartitions-operations-max-time: 0
ds-mon-lookup-operations-last-access-date: 0
ds-mon-add-operations-total-count: 0
ds-mon-remove-operations-total-time: 0
ds-mon-remove-operations-average-time: 0
ds-mon-getpartitions-operations-failed-count: 0
objectClass: ds-monitor-entry
objectClass: top
objectClass: extensibleObject
ds-mon-lookup-operations-average-time: 0
ds-mon-remove-operations-last-access-date: 0
cn: givenname
ds-mon-add-operations-last-access-date: 0
```

35.5.1.32 Monitoring a Global Index Catalog

You can use the `ldapsearch` command with the base DN `"cn=gi-catalog,cn=Global Index Catalogs,cn=monitor"` to monitor the global index catalogue.

Ensure that `givenname` corresponds to the name of the indexed attribute (for example cn, if you indexed cn), and that `gi-catalog` corresponds to the name of the global index catalog.

Run the command as follows:

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file --useSSL \
  --trustAll -b "cn=gi-catalog,cn=Global Index Catalogs,cn=monitor" \
  "(objectclass=*)"
```

Depending on your configuration, output will be similar to the following:

```
dn: cn=gi-catalog,cn=Global Index Catalogs,cn=monitor
ds-mon-replication-received-update-message-errors: 0
ds-mon-configured-index-number: 1
ds-mon-replication-full-update-pending-attribute:
ds-mon-replication-full-update-status: NONE
ds-mon-state: RUNNING_STANDALONE
ds-mon-replication-published-update-message-number: 0
ds-mon-replication-active: false
ds-mon-replication-auto-sync-retries: 0
```

```
ds-mon-replication-published-update-message-errors: 0
ds-mon-replication-full-update-errors: 0
ds-mon-replication-received-update-message-number: 0
ds-mon-replication-auto-sync-is-running: false
objectClass: ds-monitor-entry
objectClass: top
objectClass: extensibleObject
ds-mon-replication-configured: false
cn: gi-catalog
```

35.5.2 Monitoring Using the `manage-tasks` Command

Oracle Unified Directory provides a tasks back end that provides a mechanism for scheduling and processing certain tasks, such as `import-ldif`, `export-ldif`, `backup`, and `restore`. You can schedule a task to run at specific times and at recurring periods.

To monitor scheduled tasks, use the `manage-tasks` command. For more information, see Configuring Commands As Tasks.

35.5.3 Monitoring the Server Using JConsole

The JConsole (`jconsole`) Java utility is a JMX-compliant, graphical tool that connects to a running Java Virtual Machine that has been started with the management agent. This generic tool can be used to access server monitoring information.

This section contains the following topics:

- Configuring JMX on a Server Instance
- Starting JConsole
- Understanding How to Access a Server Instance From JConsole
- Viewing Monitoring Information Using JConsole

35.5.3.1 Configuring JMX on a Server Instance

To configure JMX on a server instance:

1. Start the server.
2. Enable the JMX Connection Handler and set the port number to be used with JMX.

   Choose a port that is not in use and to which the user that is running the server has access rights.

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     set-connection-handler-prop --handler-name "JMX Connection Handler" \
     --set enabled:true --set listen-port:1689
   ```

3. Add the JMX read, write, and notify privileges to the root DN.

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -X -n \
     set-root-dn-prop \
     --add default-root-privilege-name:jmx-read \
     --add default-root-privilege-name:jmx-write \
     --add default-root-privilege-name:jmx-notify
   ```

4. Restart the server.

35.5.3.2 Starting JConsole

You must start the console by typing `jconsole` in a terminal window.

To run `jconsole` from the command line, you might have to add *JAVA_HOME*/bin to your path, where `JAVA_HOME` is the directory containing the JDK. Alternatively, you can enter the full path when you type the command.

35.5.3.3 Understanding How to Access a Server Instance From JConsole

To connect JConsole to a server instance, use the Remote Process fields. The following fields are required:

- **JMX URL**:

  ```
  service:jmx:rmi:///jndi/rmi://''host'':''port''/
  org.opends.server.protocols.jmx.client-unknown
  ```

  – *host* is a host name, an IPv4 numeric host address, or an IPv6 numeric address enclosed in square brackets.

  – *port* is the decimal port number of the JMX connector. (See Configuring Alerts and Account Status Notification Handlers.)

  The default JMX URL is:

  ```
  service:jmx:rmi:///jndi/rmi://198.51.100.0:1689/
  org.opends.server.protocols.jmx.client-unknown
  ```

- **User Name.** A valid LDAP user name.

  The default Directory Manager user name is `cn=Directory Manager`.

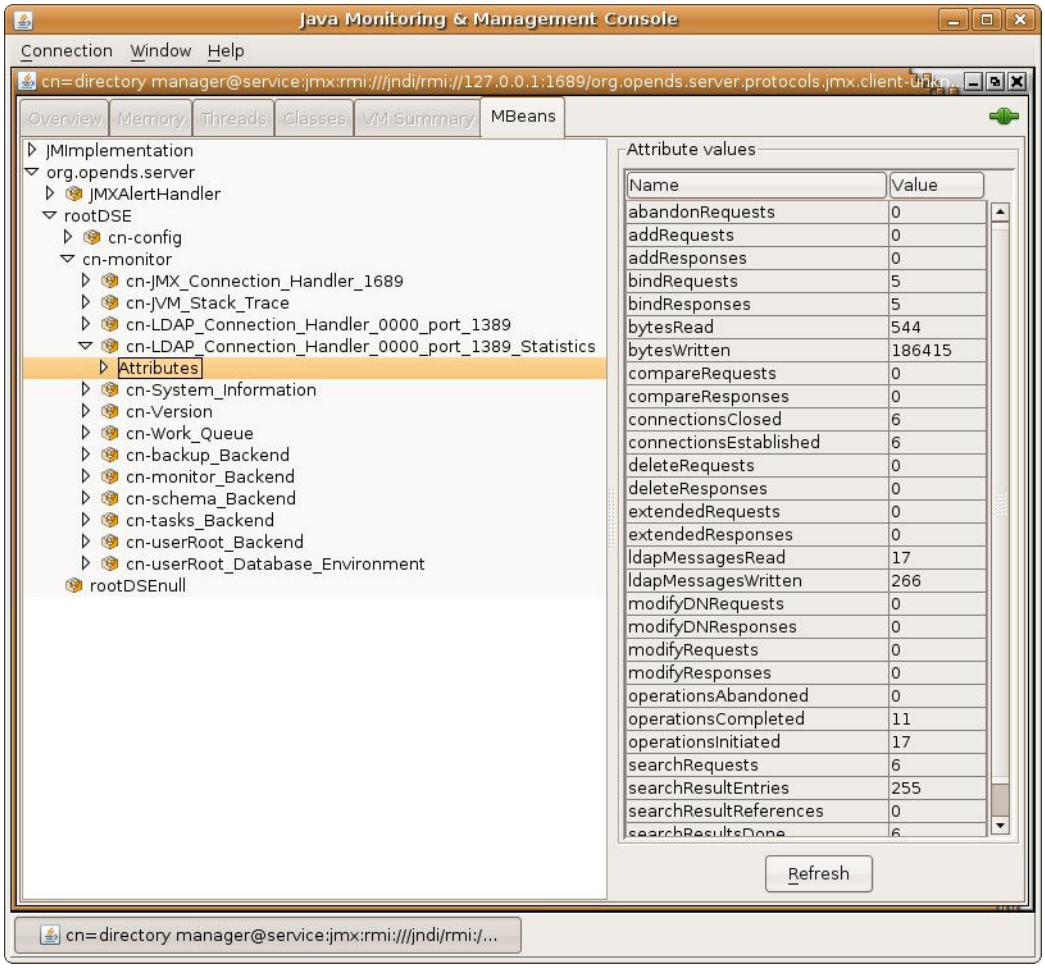
- **Password.** The user's LDAP password.

35.5.3.4 Viewing Monitoring Information Using JConsole

When JConsole is connected to a server instance, it displays management objects (MBeans). The tree on the left pane shows all MBeans currently available. You can access server monitoring information in the right hand pane by selecting the associated MBean.

The following figure shows the attribute list for a server `cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor`.

Figure 35-1 Java Monitoring and Management Console



35.5.4 Accessing Logs

The server provides logging mechanisms to record access, error, or debugging information for the server instance. Multiple loggers of a given type can be active at any time, which makes it possible to create logs for specific subtrees or different repositories. The server does not currently provide logging filters to restrict the type of information in the logs.

This section contains the following topics:

- Understanding the Different Log Types
- Viewing the Access Logs
- Viewing the Audit Logs
- Viewing the Debug Logs
- Viewing the Error Logs
- Viewing the Replication Repair Logs
- Viewing the server.out Logs
- Viewing the Setup Logs

35.5.4.1 Understanding the Different Log Types

Oracle Unified Directory supports the following logs:

- **Access logs**. Access logs record information about the types of operations processed by the directory server. Access logs are provided by default.
- **Audit logs**. Audit logs are a type of access log and record all activity on the directory server. Audit logs are not enabled by default.
- **Debug logs**. Debug logs record information that can be used for troubleshooting directory server problems or for providing detailed information about the directory server's processing. Debug logs are not enabled by default.
- **Error logs**. Error logs record all warnings, errors, or significant events that occur during directory server processing.
- **Replication repair logs**. Replication repair logs record inconsistencies on a single directory server in a topology.

  The replication repair log is read-only and its use is restricted to enabling replication conflict resolution.
- **oud-setup logs**. Setup logs record the equivalent command line arguments executed during the installation of an Oracle Unified Directory proxy server instance or replication gateway instance. This log enables you to perform a "silent install" of the proxy server or gateway server, based on a previous installation.

  This file is not output for directory server instances.
- **server.out logs**. Server.out logs record the bootstrapping configuration process, list extensions loaded from jar files, and indicate connection and alert notification activity. Currently, it is not possible to change the location where the *server.out* logs are written.

35.5.4.2 Viewing the Access Logs

To access the logs using the UNIX `cat` command:

1. Change to the logs directory of the server instance.

   ```
   $ cd INSTANCE_DIR/OUD/logs
   ```

2. Open the `access` file by using a text editor or the UNIX `cat` command.

   ```
   $ cat access | more
   [10/Jan/2012:12:02:11 +0100] CONNECT conn=0 from=198.51.100.0:55416
   to=198.51.100.0:5444 protocol=LDAPS
   [10/Jan/2012:12:02:12 +0100] BIND REQ conn=0 op=0 msgID=1 type=SIMPLE
   dn="cn=Directory Manager"
   [10/Jan/2012:12:02:12 +0100] BIND RES conn=0 op=0 msgID=1 result=0
   authDN="cn=Directory Manager,cn=Root
    DNs,cn=config" etime=36
   [10/Jan/2012:12:02:12 +0100] UNBIND REQ conn=0 op=1 msgID=2
   [10/Jan/2012:12:02:12 +0100] DISCONNECT conn=0 reason="Client Disconnect"
   ...(more output)...
   ```

35.5.4.3 Viewing the Audit Logs

To view the audit logs using the UNIX `cat` command:

1. If the audit log publisher is not already enabled, enable it as described in Enabling a Log Publisher.

2. Change to the logs directory of the server instance.

```
$ cd INSTANCE_DIR/OUD/logs
```

3. Open the `audit` file by using a text editor or the UNIX `cat` command.

```
$ cat audit | more
# 11/Jan/2012:11:20:00 +0100; conn=10; op=18
dn: cn=File-Based Audit Logger,cn=Loggers,cn=config
changetype: modify
replace: ds-cfg-enabled
ds-cfg-enabled: true
-
replace: modifiersName
modifiersName: cn=directory manager
-
replace: modifyTimestamp
modifyTimestamp: 20120111102000Z

# 11/Jan/2012:11:20:20 +0100; conn=11; op=6
dn: cn=File-Based Debug Logger,cn=Loggers,cn=config
changetype: modify
replace: ds-cfg-enabled
ds-cfg-enabled: true
-
replace: modifiersName
modifiersName: cn=directory manager
-
replace: modifyTimestamp
modifyTimestamp: 20120111102020Z
...(more output)...
```

35.5.4.4 Viewing the Debug Logs

To access the debug logs using the UNIX `cat` command:

1. If the debug log publisher is not already enabled, enable it as described in Enabling a Log Publisher.
2. Change to the logs directory of the server instance.

```
$ cd INSTANCE_DIR/OUD/logs
```

3. Open the `debug` file by using a text editor or the UNIX `cat` command.

```
$ cat debug | more
[11/Jan/2012:11:39:48 +0100] 0 caught error thread={Worker Thread 43(118)}
threadDetail={parentThread=main(1) isDaemon=false
clientConnection=LDAP client connection from 198.51.100.0:56288
to 198.51.100.0:2389 operation=SearchOperation(connID=13, opID=1,
baseDN=dc=example,dc=com, scope=wholeSubtree, filter=(objectclass=*)) }
method={run(SearchOperationBasis.java:1513)}
caught={org.opends.server.types.CanceledOperationException: Client Disconnect}
...(more output)...
```

35.5.4.5 Viewing the Error Logs

To view the error logs using the UNIX `cat` command:

1. Change to the logs directory of the server instance.

```
$ cd INSTANCE_DIR/OUD/logs
```

2. Open the `errors` file by using a text editor or the UNIX `cat` command.

```
#cat errors

[22/Jan/2015:05:54:16 -0800] category=RUNTIME_INFORMATION severity=NOTICE
 msgID=20381717 msg=Installation Directory:
/local/OUD_BASE/OracleUnifiedDirectory
[22/Jan/2015:05:54:16 -0800] category=RUNTIME_INFORMATION
severity=NOTICE msgID=20381719 msg=Instance Directory:
/local/OUD_BASE/asinst_1/OUD
[22/Jan/2015:05:54:16 -0800] category=RUNTIME_INFORMATION
severity=NOTICE msgID=20381713 msg=JVM Information: 1.7.0_67-b01 by
Oracle Corporation, 64-bit architecture, 1351614464 bytes heap size
[22/Jan/2015:05:54:16 -0800] category=RUNTIME_INFORMATION severity=NOTICE
 msgID=20381714 msg=JVM Host: host1, running
Linux 2.6.18-238.0.0.0.1.el5xen amd64, 6081740800 bytes physical memory size,
 number of processors available 2
[22/Jan/2015:05:54:16 -0800] category=RUNTIME_INFORMATION severity=NOTICE
 msgID=20381715 msg=JVM Arguments: "-Dorg.opends.server.scriptName=start-ds"
[22/Jan/2015:05:54:17 -0800] category=JEB severity=NOTICE msgID=8847402
msg=The database backend cn=virtualAcis,cn=Workflow Elements,cn=config
 containing 0 entries has started
[22/Jan/2015:05:54:17 -0800] category=JEB severity=NOTICE msgID=8847402
msg=The database backend cn=userRoot,cn=Workflow Elements,
cn=config containing 20002 entries has started
[22/Jan/2015:05:54:18 -0800] category=PROTOCOL severity=NOTICE msgID=2556180
 msg=Started listening for new connections on Administration Connector 0.0.0.0
 port 4444
[22/Jan/2015:05:54:18 -0800] category=PROTOCOL severity=NOTICE msgID=2556180
 msg=Started listening for new connections on LDAP Connection Handler 0.0.0.0
 port 1389
[22/Jan/2015:05:54:18 -0800] category=PROTOCOL severity=NOTICE msgID=2556180
 msg=Started listening for new connections on LDAP Connection Handler 0.0.0.0
 port 1636
[22/Jan/2015:05:54:18 -0800] category=CORE severity=NOTICE msgID=458887
msg=The Directory Server has started successfully
[22/Jan/2015:05:54:18 -0800] category=CORE severity=NOTICE msgID=458891
msg=The Directory Server has sent an alert notification generated by class
 org.opends.server.core.DirectoryServer (alert type org.opends.server.DirectoryServerStarted, alert
ID 458887):
The Directory Server has started successfully
```

35.5.4.6 Viewing the Replication Repair Logs

To view the replication repair logs using the UNIX `cat` command:

1. Change to the logs directory of the server instance.

   ```
   $ cd INSTANCE_DIR/OUD/logs
   ```

2. Open the `replication` file by using a text editor or the UNIX `cat` command.

   ```
   $ cat replication | more
   [13/Jan/2012:15:00:50 +0100] category=SYNC severity=NOTICE msgID=15139035
    msg=The replication server database has version 2 format
   [13/Jan/2012:15:00:50 +0100] category=SYNC severity=NOTICE msgID=15138878
    msg=Replication is up and running for domain cn=admin data with replication
    server id 18049 host1/198.51.100.0:8989 - local server id is 9338 - data
    generation is 93408 [13/Jan/2012:15:00:52 +0100] category=SYNC severity=NOTICE
    msgID=15138878 msg=Replication is up and running for domain dc=example,dc=com
    with replication server id 18049 host1/198.51.100.0:8989 - local server id is
    25340 - data generation is 19449577 [13/Jan/2012:15:00:53 +0100] category=SYNC
    severity=NOTICE msgID=15138878 msg=Replication is up and running for domain
    cn=schema with replication server id 18049 host1/198.51.100.0:8989 - local
    server id is 13881 - data generation is 8408 [13/Jan/2012:15:08:28 +0100]
   ```

```
 category=SYNC severity=NOTICE msgID=15138893 msg=On suffix cn=admin data,
 replication server 3844 presented generation ID=-1 when expected generation
 ID=93408 [13/Jan/2012:15:08:28 +0100] category=SYNC severity=MILD_ERROR
 msgID=14876753 msg=In RS 18049 for dn cn=admin data, update
 00000134d765d4b1247a00000001 will not be sent to RS 3844 with generation id
 -1 different from local generation id 93408 [13/Jan/2012:15:08:28 +0100]
 category=SYNC severity=MILD_ERROR msgID=14876753 msg=In RS 18049 for
 dn cn=admin data, update 00000134d765d4b1247a00000002 will not be sent to
 RS 3844 with generation id -1 different from local generation id 93408
...(more output)...
```

35.5.4.7 Viewing the server.out Logs

You can view the server.out logs.

Use the UNIX `cat` command to view the server.out logs:

> **See Also:**
>
> Logging OUD Instance Name in server.out Logs

1. Change to the logs directory of the server instance.

   ```
   $ cd INSTANCE_DIR/OUD/logs
   ```

2. Open the `server.out` file by using a text editor or the UNIX `cat` command.

   ```
   $ cat server.out | more
   [23/May/2011:02:27:59 -0700] category=CORE severity=INFORMATION msgID=132
     msg=The Directory Server is beginning the configuration bootstrapping process
   [23/May/2011:02:28:00 -0700] category=EXTENSIONS severity=INFORMATION msgID=1049147
     msg=Loaded extension from file '/OUD_BASE/ORACLE_HOME/lib/extensions/
   globalindex.jar'
     (build 1.0.0)
   [23/May/2011:02:28:00 -0700] category=EXTENSIONS severity=INFORMATION msgID=1049147
     msg=Loaded extension from file '/OUD_BASE/ORACLE_HOME/lib/extensions/replication-
   gateway.jar'
     (build 1.0.0)
   [23/May/2011:02:28:00 -0700] category=EXTENSIONS severity=INFORMATION msgID=1049147
     msg=Loaded extension from file '/OUD_BASE/ORACLE_HOME/lib/extensions/
   loadbalancing.jar'
     (build 1.0.0)
   [23/May/2011:02:28:00 -0700] category=EXTENSIONS severity=INFORMATION msgID=1049147
     msg=Loaded extension from file '/OUD_BASE/ORACLE_HOME/lib/extensions/
   virtualization.jar'
     (build 1.0.0)
   [23/May/2011:02:28:00 -0700] category=EXTENSIONS severity=INFORMATION msgID=1049147
     msg=Loaded extension from file '/OUD_BASE/ORACLE_HOME/lib/extensions/
   distribution.jar'
     (build 1.0.0)
   ...
   more output
   ...
   ```

35.5.4.7.1 Logging OUD Instance Name in server.out Logs

The OUD instance name is captured by default as part of the header of each log message generated by OUD.

You can view the content of the server.out log file using the `cat` command:

```
cat ../logs/server.out
[12/Feb/2024:08:38:05 +0000][asinst_1] category=CORE severity=INFORMATION
msgID=132 msg=The Directory Server is beginning the configuration
bootstrapping process
[12/Feb/2024:08:38:06 +0000][asinst_1] category=CORE severity=NOTICE
msgID=458886 msg=Oracle Unified Directory 14.1.2.1.241104 (build
20240211082854Z, R2402091051) starting up
[12/Feb/2024:08:38:08 +0000][asinst_1] category=RUNTIME_INFORMATION
severity=NOTICE msgID=20381717 msg=Installation Directory:  /scratch/
OUD_BASE/OUD/MAIN/OracleUnifiedDirectory
[12/Feb/2024:08:38:08 +0000][asinst_1] category=RUNTIME_INFORMATION
severity=NOTICE msgID=20381719 msg=Instance Directory:      /scratch/
OUD_BASE/OUD/MAIN/asinst_1/OUD
[12/Feb/2024:08:38:08 +0000][asinst_1] category=RUNTIME_INFORMATION
severity=NOTICE msgID=20381713 msg=JVM Information: 1.8.0_211-b12 by Oracle
Corporation, 64-bit architecture, 6484000768 bytes heap size
```

You can disable logging the instance name in server.out by modifying the `start-ds.java-args` property in the config/java.properties.

To disable logging the instance name in server.out, perform following steps:

1. Set the `-Dlog.instance.details` parameter to `false` in the config/java.properties file as follows:

   ```
   start-ds.java-args=-Xms6285m -Xmx6285m -d64 -XX:+UseCompressedOops -server
   -Xmn1g -XX:MaxTenuringThreshold=1 -XX:+UseConcMarkSweepGC -
   XX:CMSInitiatingOccupancyFraction=60 -Dlog.instance.details=false
   ```

2. Run the `dsjavaproperties` command to apply the new settings.

The instance name will no longer be logged when the server restarts.

> **Note:**
>
> The instance name will only be logged for logs generated by OUD. Messages from the JDK or other underlying libraries may not have the instance name logged.

35.5.4.8 Viewing the Setup Logs

Setup log files can be generated by `oud-proxy-setup`, `oud-setup`, or `oud-replication-gateway-setup`. You can view a setup log file for any kind of instance, but the output differs slightly, depending on the instance type. For example:

- Output for a Directory Server Instance

  ```
  Jan 27, 2015 4:40:04 PM org.opends.quicksetup.QuickSetupLog initLogFileHandler
  INFO: QuickSetup application January 27, 2015 4:40:04 PM MET
  ```

- Output for a Replication Gateway Instance

  ```
  Jan 27, 2015 2:53:21 PM org.opends.guitools.util.ControlPanelLog initLogFileHandler
  INFO: Application launched January 27, 2015 2:53:21 PM MET
  ```

- Output for a Proxy Server Instance

```
Jan 27, 2015 2:40:13 PM com.sun.dps.ui.deploy.SetupLog initLogFileHandler
INFO: oudproxy-setup application launched January 27, 2015 2:40:13 PM MET
```

To view a setup log:

1. Change to the `logs` directory of the server instance.

```
$ cd INSTANCE_DIR/OUD/logs
```

2. Open the `oud-proxy-setup`, `oud-setup`, or `oud-replication-gateway-setup` file by using a text editor or the UNIX `cat` command. For example, open the `oud-setup` file by typing

```
$ cat oud-setup | more
```

35.6 Monitoring the Server With SNMP

Oracle Unified Directory provides a Simple Network Management Protocol (SNMP) connection handler for Management Information Base (MIB) 2605 support. The MIB 2605 allows an SNMP manager to access the server monitoring information. The MIB contains the SNMP connection handler, the required classes to support MIB 2605 objects and SNMP requests, and the SNMP adapter that allows an SNMP manager to access the server monitoring information. The SNMP MIB 2605 description is stored in a file located in `install-dir/snmp/mib/rfc2605.txt`.

Oracle Unified Directory allows you to enable and configure the SNMP connection handler.

Before you start on the procedures in this section, ensure that you have set up an SNMP-managed network for your particular system.

This section contains the following topics:

35.6.1 Configuring SNMP in the Server

You can configure Oracle Unified Directory for monitoring through the Simple Network Management Protocol (SNMP). The server uses the Java Dynamic Management Kit (JDMK) to create smart agents for the SNMP connection handler.

1. Verify that the SNMP connection handler is displayed under the list of current connection handlers by using `dsconfig` as follows.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  list-connection-handlers

Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
LDIF Connection Handler  : ldif : true    : -           : -
SNMP Connection Handler  : snmp : false   : 161         : -
```

2. Use the `dsconfig` command to enable SNMP for the server and to set the listen port.

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j pwd-file -n -X \
  set-connection-handler-prop --handler-name "SNMP Connection Handler" \
  --set enabled:true --set listen-port:8085
```

35.6.2 Viewing the SNMP Connection Handler Properties

You use the `dsconfig` command to view the properties of an existing SNMP connection handler.

Run the command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -j pwd-file -n \
  get-connection-handler-prop --handler-name "SNMP Connection Handler"
```

The connection handler properties are listed with their values, as follows.

```
Property            : Value(s)
--------------------:-----------------------------------------
allowed-client      : -
allowed-manager     : *
allowed-user        : *
community           : OUD
denied-client       : -
enabled             : false
listen-port         : 161
opendmk-jarfile     : -
registered-mbean    : false
security-agent-file : config/snmp/security/oud-snmp.security
security-level      : authnopriv
trap-port           : 162
traps-community     : OUD
traps-destination   : -
```

35.6.3 Accessing SNMP on a Server Instance

You can check the status of the SNMP Connection Handler on a server instance.

To access SNMP on a server instance:

1. Restart the server by using `stop-ds` and `start-ds`.

   If the server was started and no modifications were made to the configuration, the restart operation is not required.

2. Check that the SNMP Connection Handler is running.

```
$ snmpwalk -v 2c -c OUD@OUD localhost:161 mib-2.66
SNMPv2-SMI::mib-2.66.1.1.1.1 = STRING: "Oracle Unified Directory Server
11.1.2.2.0 - 20131010000044Z"
SNMPv2-SMI::mib-2.66.1.1.2.1 = STRING: "INSTANCE_DIR/bin"
SNMPv2-SMI::mib-2.66.1.1.3.1 = Gauge32: 35
SNMPv2-SMI::mib-2.66.1.1.4.1 = Gauge32: 1
SNMPv2-SMI::mib-2.66.1.1.5.1 = Gauge32: 0
SNMPv2-SMI::mib-2.66.1.1.6.1 = Counter32: 0
SNMPv2-SMI::mib-2.66.1.1.7.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.1.1.1 = INTEGER: 1
SNMPv2-SMI::mib-2.66.2.1.1.1.2 = INTEGER: 2
SNMPv2-SMI::mib-2.66.2.1.1.1.3 = INTEGER: 3
SNMPv2-SMI::mib-2.66.2.1.2.1.1 = OID: SNMPv2-SMI::internet.27.3.8085
SNMPv2-SMI::mib-2.66.2.1.2.1.2 = OID: SNMPv2-SMI::internet.27.3.1389
```

```
SNMPv2-SMI::mib-2.66.2.1.2.1.3 = OID: SNMPv2-SMI::enterprises.42
SNMPv2-SMI::mib-2.66.2.1.3.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.3.1.2 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.3.1.3 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.2 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.3 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.5.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.5.1.2 = Counter32: 1
...
```

The managed objects included in the MIB 2605 are divided into three tables: `dsTable`, `dsAppliIfOpsTable`, and `dsIntTable`. Currently, the `dsIntTable` table is not implemented.

35.6.4 Understanding SNMP Security Configuration

The security configuration of Simple Network Management Protocol (SNMP) depends on the version of SNMP you are using.

The following topics describe the security configuration for SNMP V1 and V2c, and V3:

- About SNMP Security Configuration: V1 and V2c
- About SNMP Security Configuration: V3
- About SNMP USM Configuration: V3

35.6.4.1 About SNMP Security Configuration: V1 and V2c

Under SNMP v1 and SNMP v2c, agents act as information servers, and the IP-based access control protects this information from unauthorized access. By default, the MIB 2605 is accessible in v1 and v2c by using the community string `OUD@OUD`. All managers are allowed to read the monitoring information exposed by the MIB 2605.

> **Note:**
>
> Only read access is authorized on the MIB 2605.

You can configure SNMP v1 and SNMP v2c by setting the SNMP connection handler properties with the `dsconfig` command. Properties related to the SNMP v1 and SNMP v2c security configuration include:

- `allowed-manager`
- `community`

SNMP v1 traps are sent on server startup and server shutdown. By default, these traps are sent to `localhost` and use the trap community string "`OUD`".

> **Note:**
>
> The default trap port might have to be changed to a value that is allowed by the system.

SNMP traps are also configured by setting the SNMP connection properties with the `dsconfig` command. Properties related to SNMP traps include:

- `trap-port`
- `traps-community`
- `traps-destination`

The ACL file that corresponds to the default values of the SNMP connection handler would be represented as follows:

```
acl = {
{
communities = OUD
access = read-only
managers = all
}
}
trap = {
{
traps-community = OUD
hosts = localhost
}
}
```

35.6.4.2 About SNMP Security Configuration: V3

The SNMP v3 protocol provides more sophisticated security mechanisms than SNMP v1 and SNMP v2c. SNMP v3 implements a user-based security model (USM) that authenticates and encrypts the requests sent between agents and their managers, and provides user-based access control. A `defaultUser` template is provided for adding authorized users in the agent engine using the SNMP cloning mechanism.

Under SNMP v3, the community string described in the previous section is used as the "context" from which the MIB 2605 is registered. By default, the MIB2605 is accessible in v3 by using the context "`OUD`". All users have access to it.

The SNMP v3 UACL is configured by setting the SNMP connection handler properties with the `dsconfig` command-line utility. The properties related to SNMP v3 UACL configuration include:

- `community`
- `allowed-user`
- `security-level`

The UACL file corresponding to the default values of the SNMP connection handler would be represented as follows:

```
uacl = {
{
context-names = OUD
access = read-only
security-level = authNoPriv
```

```
users = *
}
}
```

35.6.4.3 About SNMP USM Configuration: V3

The USM MIB (that is, the MIB that defines allowed users) is registered in the `null` context and only a `snmpAdmin` user with a security level `authNoPriv` has `read-write` access to it. This `snmpAdmin` user can add additional users who can access the MIB 2605 information.

The SNMP v3 USM configuration is read from a template file that is located at `INSTANCE_DIR/OUD/config/snmp/security/oud-snmp.security`. The template file is not encrypted.

To access the MIB 2605 in the server agent, use the SNMP clone mechanism to add a user in the security file. Use `snmpAdmin` to send the SNMP request for the clone mechanism as shown here. The user to clone is `defaultUser`. The `snmpAdmin` and `defaultUser` users cannot access the MIB 2605 information.

- Admin User to add and configure other users.

  ```
  userEntry=localEngineID,snmpAdmin,null,usmHMACMD5AuthProtocol,passadmin
  ```

- Template user to be cloned with no read or write access.

  ```
  userEntry=localEngineID,defaultUser,,usmHMACMD5AuthProtocol,password,,,3,true
  ```

> **Note:**
>
> The security file is also used to make the users persistent.

35.6.5 Configuring SNMP Traps

You can configure SNMP traps for monitoring.

Existing OUD alerts are available as SNMP V1 traps. If SNMP handler is enabled, it will send the appropriate SNMP trap whenever the OUD server generates an alert. The underlying alert type and its associated message are both contained in the SNMP trap.

You can configure the following SNMP trap settings based on your requirement:

enabled-traps

Specifies the names of the server alert types that are enabled to be sent as traps. Only those server alerts are sent as traps, if this attribute has any values (unless they are also included in the disabled traps). Any server alert type that is not on the list of disabled traps will be sent as a trap if this attribute has no values.

```
dsconfig set-connection-handler-prop --handler-name "SNMP Connection Handler"
--set
      enabled-traps:org.opends.server.DirectoryServerShutdown
```

> **Note:**
>
> You can use the `--add` option for subsequent additional values. See dsconfig.

disabled-traps

Specifies the names of the server alert types that will not be sent as traps. If this attribute has any values, no traps will be sent for those server alert types. If this attribute has no values, only traps based on the `enabled-traps` configuration will be sent.

```
dsconfig set-connection-handler-prop --handler-name "SNMP Connection Handler"
--set
      disabled-traps:org.opends.server.authorization.dseecompat.AciModified
```

> **Note:**
>
> You can use the `--add` option for subsequent additional values. See dsconfig.

disable-all-traps

When this is set to `true`, the server will not send any traps.

```
dsconfig set-connection-handler-prop --handler-name "SNMP Connection Handler"
--set disable-all-traps:true
```

35.6.6 Supported SNMP Traps OID Mapping

When an alert type event occurs in the system, the OUD server sends an alert message, and the SNMP handler, if enabled, sends an associated SNMP trap.

The supported SNMP traps are listed in the following table, along with their OID mapping.

SNMP Trap	Description	OID Mapping
DirectoryServerStarted	SNMP trap for `org.opends.server.DirectoryServerStarted` alert.	1.3.6.1.4.1.111.9118.2.1
DirectoryServerShutdown	SNMP trap for `org.opends.server.DirectoryServerShutdown` alert.	1.3.6.1.4.1.111.9118.2.2
UncaughtException	SNMP trap for `org.opends.server.UncaughtException` alert.	1.3.6.1.4.1.111.9118.2.3
CannotCopySchemaFiles	SNMP trap for `org.opends.server.CannotCopySchemaFiles` alert.	1.3.6.1.4.1.111.9118.2.4
CannotWriteNewSchemaFiles	SNMP trap for `org.opends.server.CannotWriteNewSchemaFiles` alert.	1.3.6.1.4.1.111.9118.2.5

SNMP Trap	Description	OID Mapping
AciModified	SNMP trap for `org.opends.server.authorization.dseecompat.AciModified` alert.	1.3.6.1.4.1.111.9118.2.6
ACIParseFailed	SNMP trap for `org.opends.server.authorization.dseecompat.ACIParseFailed` alert.	1.3.6.1.4.1.111.9118.2.7
BackendRunRecovery	SNMP trap for `org.opends.server.BackendRunRecovery` alert.	1.3.6.1.4.1.111.9118.2.8
LDIFBackendCannotWriteUpdate	SNMP trap for `org.opends.server.LDIFBackendCannotWriteUpdate` alert.	1.3.6.1.4.1.111.9118.2.11
LDIFConnectionHandlerParseError	SNMP trap for `org.opends.server.LDIFConnectionHandlerParseError` alert.	1.3.6.1.4.1.111.9118.2.12
LDIFConnectionHandlerIOError	SNMP trap for `org.opends.server.LDIFConnectionHandlerIOError` alert.	1.3.6.1.4.1.111.9118.2.13
UniqueAttributeSynchronizationConflict	SNMP trap for `org.opends.server.UniqueAttributeSynchronizationConflict` alert.	1.3.6.1.4.1.111.9118.2.14
UniqueAttributeSynchronizationError	SNMP trap for `org.opends.server.UniqueAttributeSynchronizationError` alert.	1.3.6.1.4.1.111.9118.2.15
Replication Server UnresolvedConflict	SNMP trap for `org.opends.server.replication.UnresolvedConflict` alert.	1.3.6.1.4.1.111.9118.2.17
AccessControlDisabled	SNMP trap for `org.opends.server.AccessControlDisabled` alert.	1.3.6.1.4.1.111.9118.2.20
AccessControlEnabled	SNMP trap for `org.opends.server.AccessControlEnabled` alert.	1.3.6.1.4.1.111.9118.2.21
CannotRenameCurrentTaskFile	SNMP trap for `org.opends.server.CannotRenameCurrentTaskFile` alert.	1.3.6.1.4.1.111.9118.2.22
CannotRenameNewTaskFile	SNMP trap for `org.opends.server.CannotRenameNewTaskFile` alert.	1.3.6.1.4.1.111.9118.2.23
CannotScheduleRecurringIteration	SNMP trap for `org.opends.server.CannotScheduleRecurringIteration` alert.	1.3.6.1.4.1.111.9118.2.24
CannotWriteConfig	SNMP trap for `org.opends.server.CannotWriteConfig` alert.	1.3.6.1.4.1.111.9118.2.25

SNMP Trap	Description	OID Mapping
EnteringLockdownMode	SNMP trap for `org.opends.server.EnteringLockdownMode` alert.	1.3.6.1.4.1.111.9118.2.26
LeavingLockdownMode	SNMP trap for `org.opends.server.LeavingLockdownMode` alert.	1.3.6.1.4.1.111.9118.2.27
ManualConfigEditHandled	SNMP trap for `org.opends.server.ManualConfigEditHandled` alert.	1.3.6.1.4.1.111.9118.2.28
ManualConfigEditLost	SNMP trap for `org.opends.server.ManualConfigEditLost` alert.	1.3.6.1.4.1.111.9118.2.29
CannotWriteTaskFile	SNMP trap for `org.opends.server.CannotWriteTaskFile` alert.	1.3.6.1.4.1.111.9118.2.30
LDAPHandlerDisabledByConsecutiveFailures	SNMP trap for `org.opends.server.LDAPHandlerDisabledByConsecutiveFailures` alert.	1.3.6.1.4.1.111.9118.2.31
LDAPHandlerUncaughtError	SNMP trap for `org.opends.server.LDAPHandlerUncaughtError` alert.	1.3.6.1.4.1.111.9118.2.32
HTTPHandlerDisabledByConsecutiveFailures	SNMP trap for `org.opends.server.HTTPHandlerDisabledByConsecutiveFailures` alert.	1.3.6.1.4.1.111.9118.2.33
HTTPHandlerUncaughtError	SNMP trap for `org.opends.server.HTTPHandlerUncaughtError` alert.	1.3.6.1.4.1.111.9118.2.34
SaturationLoadBalancer	SNMP trap for `com.sun.dps.server.SaturationLoadBalancer` alert.	1.3.6.1.4.1.111.9118.2.35
UnsupportedDirectoryBackend	SNMP trap for `com.sun.dps.server.distribution.globalindex.UnsupportedDirectoryBackend` alert.	1.3.6.1.4.1.111.9118.2.36
LDAPServerExtensionUp	SNMP trap for `com.sun.dps.server.workflowelement.proxyldap.LDAPServerExtension.LDAPServerExtensionUp` alert.	1.3.6.1.4.1.111.9118.2.37
LDAPServerExtensionDown	SNMP trap for `com.sun.dps.server.workflowelement.proxyldap.LDAPServerExtension.LDAPServerExtensionDown` alert.	1.3.6.1.4.1.111.9118.2.38
FailoverLoadBalancer	SNMP trap for `com.sun.dps.server.FailoverLoadBalancer` alert.	1.3.6.1.4.1.111.9118.2.40

35.7 Monitoring a Replicated Topology

When directory server replication is enabled, changes made on one directory server are immediately propagated, or *replicated*, to multiple different directories in the topology. You can monitor Oracle Unified Directory replication status by using the `dsreplication status` command to obtain replication status information.

If you enable a replication gateway server, you can monitor replication status for both Oracle Unified Directory and ODSEE directory servers in the topology.

For general information about how directory server replication works, see Understanding the Oracle Unified Directory Replication Model. For general information about using a replication gateway, see Overview of the Replication Gateway.

This section contains the following subsections:

- Monitoring Basic Oracle Unified Directory Replication Status Using `dsreplication`
- Monitoring Advanced Oracle Unified Directory Replication Status Using `dsreplication`
- Monitoring Oracle Unified Directory and ODSEE Replication Status in Deployments Using Replication Gateways

35.7.1 Monitoring Basic Oracle Unified Directory Replication Status Using `dsreplication`

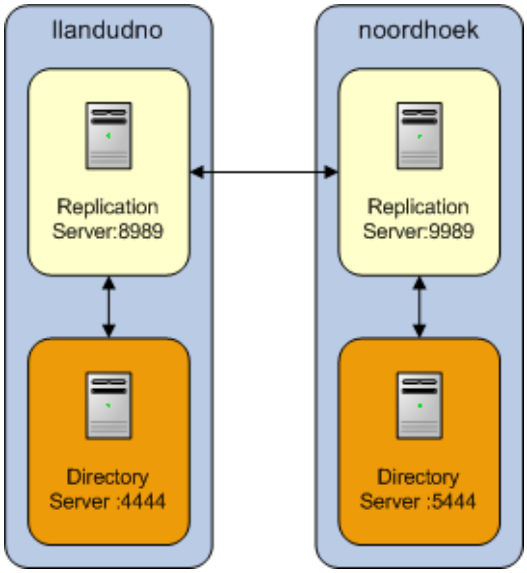
The simplest way to monitor replication on Oracle Unified Directory is to use the `dsreplication status` command.

This command provides a tabular view of the replication status, including the following information:

- The topology and its connections
- The latency between replicated servers
- The data consistency across replicated servers
- The security configuration between replicated servers
- The replication protocol peer to peer

The examples in the remainder of this section assume the following simple replication topology.

Figure 35-2 Simple Replication Topology



The following subsections describe how to retrieve the basic replication status information:

- Viewing Minimal Basic Replication Status Information
- Viewing Additional Basic Replication Status Information

For information about retrieving more in-depth information, see Monitoring Advanced Oracle Unified Directory Replication Status Using `dsreplication` .

35.7.1.1 Viewing Minimal Basic Replication Status Information

You use the `dsreplication` command to view the minimal basic replication status information.

Run the command as follows:

```
$ dsreplication status --adminUID admin --adminPasswordFile pwd.txt -X \
 --hostname host1 --port 4444
```

The following information is displayed:

- **Server.** Lists the LDAP servers in the topology and the port on which they are listening for LDAP connections.
- **Entries.** Indicates the number of entries on each server for the specified base DN. If the information in this column is different across all the servers, the replication topology is not synchronized.
- **M.C.** Indicates the number of updates already pushed by the other LDAP servers in the topology, but not yet replayed on the specified LDAP server. If this number is high on a particular server, investigate the latency of that server.
- **A.O.M.C.** Specifies the approximate date of the oldest update pushed by the other directory servers in the topology, but not yet processed on the specified LDAP server.
- **Port.** Indicates the port of the replication server (if any) that is configured in the instance. Usually the LDAP servers in the instance are connected to it.
- **Status.** Indicates the status of the replication domain on this directory server.

For directory servers that contain data (replication domains), the status can be one of the following:

- **Normal.** The connection to a replication server is established with the correct data set. Replication is working. If assured mode is used, then acknowledgments from this directory server are sent.
- **Late.** The connection to a replication server is established with the correct data set. Replication is marked Late when the number of missing changes in the directory server exceeds the threshold defined in the replication server configuration. When the number of changes goes below this threshold, the status will go back to Normal.
- **Full Update.** The connection to a replication server is established and a new data set is received from this connection (online import), to initialize the local back end.
- **Bad Data Set.** The connection to a replication server is established with a data set that is different from the rest of the topology. Replication is not working. Either the other directory servers of the topology should be initialized with a compatible data set, or this server should be initialized with another data set that is compatible with the other servers.
- **Not Connected.** The directory server is not connected to any replication server.
- **Unknown.** The status cannot be determined. This occurs mainly when the server is down or unreachable but it is referenced in the monitoring of another server.
- **Invalid.** This is for internal use. If the directory server changes its state and the transition is impossible according to state machine, the INVALID_STATUS is returned.

When a directory server such as a replication server does not contain replicated data, or when you specify the `--expanded option`, the replication server status can have the following values:

- **Up.** The replication server is up and running and is connected properly to the other servers.
- **Down.** The replication server is not connected to other servers and is not running properly.
- **Unknown.** The status cannot be determined. This occurs mainly when the Oracle Unified Directory instance where the replication server is down or unreachable but the replication server is referenced in the configuration of another server.

35.7.1.2 Viewing Additional Basic Replication Status Information

You use the `dsreplication` command to view the additional basic replication status information.

Run the command as follows:

```
$ dsreplication status --adminUID admin --adminPasswordFile pwd.txt -X \
 --hostname host1 --port 4444 --dataToDisplay compat-view
```

The resulting `compat-view` is the same view that was displayed in previous versions of Oracle Unified Directory. In addition to the information described in Viewing Minimal Basic Replication Status Information, the following information is also displayed:

- **Encryption.** Indicates whether SSL encryption is enabled between the LDAP server and its replication server.
- **Trust.** Indicates whether this server is configured as a trusted or untrusted server. For more information, see Understanding Isolated Replicas.

- **U.C.** Specifies the number of changes that have been made on an untrusted server, and not yet replicated to the topology. For more information, see Understanding Isolated Replicas.
- **Change Log**. Indicates whether the external change log is enabled for the base DN on this server. For more information, see Using the External Change Log.
- **Group ID**. The ID of the replication group to which the server belongs. For more information, see About Replication Groups.
- **Connected To.** Displays the name, IP address and replication port of the replication server to which this directory server is connected.

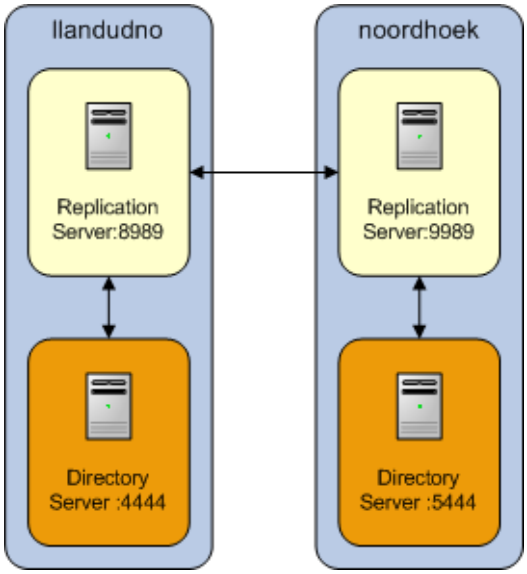
35.7.2 Monitoring Advanced Oracle Unified Directory Replication Status Using `dsreplication`

You can use the `dsreplication enable` command and its `dataToDisplay` option to track specific monitoring attributes. This provides you a more in-depth and comprehensive view of the replication status than the basic replication status information.

Monitoring information is consolidated by replication servers. Therefore, monitoring information can only be retrieved by searching a directory server that hosts a running replication server.

The examples in the remainder of this section assume the following simple replication topology.

Figure 35-3 Simple Replication Topology



This section covers the following monitoring topics:

- Viewing a Comprehensive List of Available Replication Status Information
- Monitoring the Topology and Its Connections
- Monitoring Replication Latency
- Monitoring Data Consistency
- Monitoring Replication Security

- Monitoring Replicated Updates
- Monitoring Replication Conflicts

35.7.2.1 Viewing a Comprehensive List of Available Replication Status Information

You use the `dsreplication` command to view a list of all replication status attributes that can be displayed, including a short description for attribute.

Run the command as follows:

```
dsreplication status --advanced --listDataToDisplay
```

35.7.2.2 Monitoring the Topology and Its Connections

Each directory server contains a list of candidate replication servers for each replicated base DN. However, a directory server is *connected* to only one replication server at a time.

To obtain an overview of the replication topology and its connections, run the following command on any directory server in the topology that hosts a replication server:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt
-n --dataToDisplay connected-to --dataToDisplay lost-connections
Establishing connections ...... Done.

dc=example,dc=com - Replication Enabled
=======================================
Server             : Port [1] : Connected To [2]   : L.C. [3]
-------------------:----------:--------------------:---------
host1:4444         : 8989     : host1:8989 (GID=1) : 0
host2:5444         : 9989     : host2:9989 (GID=1) : 0

[1] The replication port used to communicate between the servers whose
contents are being replicated.
[2] The replication server this element is connected to with its group
ID between brackets.
[3] Lost Connections.
```

The `Connected To` column indicates the replication server to which each directory server is currently connected for a particular base DN. Because all replication servers are permanently connected to all other replication servers, the `Connected To` column does not list replication servers.

The Lost Connections (`L.C.`) column indicates the number of connection breaks between directory servers and replication servers. The value indicated for each directory server should be close to the number of times that replication has been stopped on that server. If the value of this attribute is much higher, there are unexpected connection losses that must be investigated.

35.7.2.3 Monitoring Replication Latency

Monitoring replication latency enables you to establish whether a specific replication server is lagging behind other servers in the topology. This provides a complete view of any replication delays and the current quality of service.

To monitor replication latency, run the following search on any server in the topology that hosts a replication server:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt -n --
dataToDisplay missing-changes --dataToDisplay aoomc
```

```
Establishing connections ..... Done.

dc=example,dc=com - Replication Enabled
========================================
Server             : M.C. [1] : A.O.M.C. [2] : Port [3]
-------------------:----------:--------------:---------
host1:4444         : 0        : N/A          : 8989
host2:5444         : 0        : N/A          : 9989

[1] The number of changes that are still missing on this element (and that have been
applied to at least one other server).
[2] Age of oldest missing change: the age (in seconds) of the oldest change that has not
yet arrived on this element.
[3] The replication port used to communicate between the servers whose contents are
being replicated.
```

In this example, the age of oldest missing change (`A.O.M.C.`) is expressed as the number of seconds since the command was run and the oldest update was pushed by the other directory servers in the topology. The oldest update may not yet be processed on the specified directory server.

The Missing Changes (`M.C.`) column specifies the number of updates already pushed by the other directory servers in the topology, but not yet replayed on the specified directory server.

> **Note:**
>
> If the replication latency, as defined by these attributes, is high, look at the number of updates sent and received to identify the servers in the topology that are causing the latency. These attributes are described later in this document.

35.7.2.4 Monitoring Data Consistency

Monitoring data consistency enables you to establish whether each replication server in the topology is synchronized and up-to-date with the latest changes that have occurred in the topology.

If data is not consistent, `Bad Data Set` is indicated in the `Status` column. To see the generation IDs, run the following command:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt -n --
dataToDisplay status --dataToDisplay generation-id
Establishing connections ......... Done.

dc=example,dc=com - Replication Enabled
========================================
Server             : Port [1] : Status [2]   : Gen. ID [3]
-------------------:----------:--------------:------------
host1:4444         : 8989     : Bad Data Set : 19399981
host2:5444         : 9989     : Normal       : 19399981

[1] The replication port used to communicate between the servers whose contents are
being replicated.
[2] The status of the replication on this element.
[3] The generation ID: the version of the data in each replicated base DN, for each
directory server.
```

The Generation ID (`Gen. ID`) column indicates the *version* of the data in each replicated base DN, for each directory server. Notice that the generation ID on all servers for the base DN `dc=example,dc=com` is `19399981`. The consistency of the Generation IDs means that the data on those servers is the same for that base DN.

Each directory server is also aware of the Generation ID of the replication server to which it is connected. The Generation ID of a replication server relates to the updates that are stored in its change log database for that base DN.

Replication is considered to be working correctly between two directory servers, for a specified base DN, when those servers and their replication server all have the same generation ID.

35.7.2.5 Monitoring Replication Security

A secure replication topology has SSL encryption enabled between servers, for a particular base DN.

To monitor replication security, run the following command on any server in the topology that hosts a replication server:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt -n --
dataToDisplay secure-conf
Establishing connections ......... Done.

dc=example,dc=com - Replication Enabled
=======================================
Server             : Port [1] : Encryption [2]
-------------------:----------:---------------
host1:4444         : 8989     : Disabled
host2:5444         : 9989     : Disabled

[1] The replication port used to communicate between the servers whose contents are
being replicated.
[2] Whether the replication communication initiated by this element is encrypted or not.
```

The `Encryption` column indicates whether the SSL protocol is enabled or disabled between two servers for a specified base DN. This information is available for each directory server or replication server. Authentication of replication sessions is not monitored.

You can configure the servers to use an encrypted communication using `dsreplication enable` interactively, or using the following two arguments:

`--secureReplication1`
Specifies whether the replication communication established from the first server is encrypted or not. This option will only be taken into account the first time replication is configured on the first server

`--secureReplication2`
Specifies whether the replication communication established from the second server is encrypted or not. This option will only be taken into account the first time replication is configured on the second server.

35.7.2.6 Monitoring Replicated Updates

Monitoring the number of updates that have been sent and received by the servers in a topology provides an indication of how well replication is working.

To monitor sent and received updates, run the following command:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt
-n --dataToDisplay sent-updates --dataToDisplay received-updates
--dataToDisplay send-window
Establishing connections ......... Done.

dc=example,dc=com - Replication Enabled
=======================================
Server             : Port [1] : R.U. [2] : S.U. [3] : S.W. [4]
-------------------:----------:----------:----------:---------
host1:4444         : 8989     : 0        : 0        : 100
host2:5444         : 9989     : 0        : 0        : 100

[1] The replication port used to communicate between the servers whose contents are
being replicated.
[2] Received updates.
[3] Sent updates.
[4] Send window between this element and the replication server it is connected to.
```

The Send Updated (`S.U.`) column indicates the number of updates that have been sent by this directory server or replication server.

The Received Updates (`R.U.`) column indicates the number of updates that have been received by this directory server or replication server.

The values of these attributes assist in determining the flow of updates within a topology. When replication appears to be very slow, it is helpful to monitor these attributes. If the number of updates sent by one server is consistently much higher than the number of updates received by another server, it is likely that the second server is a bottleneck in the topology.

The replication protocol controls the flow of updates between two servers. This ensures that when a high number of updates is exchanged between two servers, the servers are not prevented from processing operations with a higher priority. This functionality relies on a window mechanism where the recipient server periodically provides the sending server with the number of updates that the sending server can send.

You can specify the size of the send and receive windows, by setting the `max-send-window` and `max-rcv-window` configuration attributes. For more information, see Modifying the Replication Configuration With `dsconfig`.

35.7.2.7 Monitoring Replication Conflicts

When multiple operations are performed on the same entry at the same time, replication conflicts can occur. In some cases, the replication mechanism can resolve these conflicts. In other cases, manual conflict resolution is required.

Three types of conflict attributes can be monitored:

- `unresolved-naming-conflicts`. Indicates the number of naming conflicts that could not be resolved by the replication mechanism.
- `resolved-naming-conflicts`. Indicates the number of naming conflicts that have been resolved.
- `resolved-modify-conflicts`. Indicates the number of modify conflicts that have been resolved.

To monitor resolved and unresolved replication conflicts, run the following command:

```
bin/dsreplication status -X -p 4444 --adminPasswordFile /tmp/password.txt -n
 --dataToDisplay resolved-naming-conflicts --dataToDisplay
 unresolved-naming-conflicts --dataToDisplay resolved-modify-conflicts
```

```
Establishing connections ...... Done.

dc=example,dc=com - Replication Enabled
=======================================
Server             : Port [1] : R.N.C. [2] : U.N.C. [3] : R.M.C. [4]
-------------------:----------:------------:------------:-----------
host1:4444         : 8989     : 0          : 0          : 0
host2:5444         : 9989     : 0          : 0          : 0

[1] The replication port used to communicate between the servers whose contents are
being replicated.
[2] Resolved Naming Conflicts.
[3] Unresolved Naming Conflicts.
[4] Resolved Modify Conflicts.
```

35.7.3 Monitoring Oracle Unified Directory and ODSEE Replication Status in Deployments Using Replication Gateways

Areplication gateway is a server that translates and propagates replication information among Oracle Directory Server Enterprise Edition servers and Oracle Unified Directory servers in a replicated topology.

Translations are managed as needed, without storing any data on disk. When a replication gateway is deployed, you can use the Oracle Unified Directory `dsreplication` command or the ODSEE console to monitor replication status information.

For general information about using a replication gateway, see Overview of the Replication Gateway.

This section contains the following topics:

- Using `dsreplication` to Monitor Changes Made on the Oracle Unified Directory Topology
- Understanding How to Use DSCC to Monitor a Replication Gateway

35.7.3.1 Using `dsreplication` to Monitor Changes Made on the Oracle Unified Directory Topology

You can use `dsreplication` to monitor how changes made on the Oracle Unified Directory topology are propagated through the replication gateway to the ODSEE topology.

The following example illustrates how to monitor sent and received updates on the Oracle Unified Directory topology. Figure 35-4 shows the results returned when the following command is run:

```
# dsreplication status -d compat-view
```

Figure 35-4 Results for `dsreplication status` with a Replication Gateway Deployed

```
dc=com - Replication Enabled
============================
Server         : Entries : M.C. [1] : A.O.M.C. [2] : Port [3] : SSL [4]  : Trust [5] : U.C. [6] : Status [7] : ChangeLog [8] : Group ID [9] : Connected To [10]
---------------:---------:----------:-------------:---------:----------:-----------:----------:------------:---------------:--------------:-----------------------
localhost:4444 : 20000   : 0        : N/A          : 4989    : Disabled : Trusted   : --       : Normal     : Enabled       : 1            : localhost:4989 (GID=1)
localhost:5444 : 20000   : 0        : N/A          : 5989    : Disabled : Trusted   : --       : Normal     : Enabled       : 1            : localhost:5989 (GID=1)
localhost:6444 : 20000   : 0        : N/A          : 6989    : Disabled : Trusted   : --       : Normal     : Enabled       : 1            : localhost:6989 (GID=1)

Replication Gateway : DSEE Server    : Protocol [11] : R.O.D [12] : M.C. [1] : A.O.M.C. [2] : SSL [4]  : Status [7] : Group ID [9] : Connected To [10]
--------------------:---------------:---------------:------------:----------:-------------:----------:------------:--------------:-----------------------
localhost:8444      : localhost:2389 : Clear         : Yes        : 0        : N/A         : Disabled : Normal     : 1            : localhost:5989 (GID=1)
localhost:9444      : localhost:1389 : Clear         : Yes        : 0        : N/A         : Disabled : Normal     : 1            : localhost:4989 (GID=1)
```

These results are explained in the additional information also returned by the command:

```
[1] The number of changes that are still missing on this element (and that have been
applied to at least one other server).
[2] Age of oldest missing change: the age (in seconds) of the oldest change that has not
yet arrived on this element.
[3] The replication port used to communicate between the servers whose contents are
being replicated.
[4] Whether the replication communication initiated by this element is encrypted or not.
[5] Whether the directory server is trusted or not. Updates coming from an untrusted
server are discarded and not propagated.
[6] The number of untrusted changes. These are changes generated on this server while it
is untrusted.
    Those changes are not propagated to the rest of the topology but are effective on
the untrusted server.
[7] The status of the replication on this element.
[8] Whether the external change log is enabled for the base DN on this server or not.
[9] The ID of the replication group to which the server belongs.
[10] The replication server this element is connected to with its group ID between
brackets.
[11] The protocol used by the replication gateway to connect to the DSEE server.
[12] Replicate OUD Changes to DSEE
```

35.7.3.2 Understanding How to Use DSCC to Monitor a Replication Gateway

DSEE 6.x and ODSEE 11*g* directory servers provide a monitoring tool within Directory Service Control Center (DSCC). You can configure an Oracle Unified Directory replication gateway server to work with the DSCC and its related tool `dsccmon`, which enables you to monitor changes that have been made on the ODSEE servers and replicated to the Oracle Unified Directory topology.

Once you have installed and configured the replication gateway, the DSCC displays the following information in the **Directory Servers** panel:

- In the **Servers** tab, the replication gateway is displayed as an ODSEE server. The Description field indicates that the ODSEE server is theOracle Unified Directory replication gateway, and provides the real version of the replication gateway server. The port number, the instance path, and status of the server are also displayed.
- In the **Suffixes** tab, the replication gateway is displayed with no entries and no replication agreement. This indicates the Oracle Unified Directory topology access point. Here you can monitor the state of the Oracle Unified Directory topology and the changes done on the ODSEE servers.
- In the **Replication Agreements** tab, the replication gateway is one of the destination servers. After the replication gateway has been set up, replication monitoring begins when at least one change has been done on the ODSEE topology.
- The replication gateway is also displayed in the View Topology drawing.

Important: While DSCC enables you to view the Oracle Unified Directory replication gateway, it *does not* enable you to perform administrative operations such as starting stopping, or configuring the Oracle Unified Directory replicating gateway server.

For information about setting up the Replication Gateway, see the Installation Guide.

35.8 Monitoring the Proxy LDAP Connector

The Oracle Unified Directory proxy server uses LDAP connectors (also known as the `LDAPServerExtension` configuration object) to communicate with remote LDAP servers. Each LDAP connector manages a connection pool that can be monitored with a real-time monitoring panel.

This monitoring panel reports the following information:

- Server status
- Current throughput for each operation type
- Connection pool status

This section contains the following topics:

- Displaying the Monitoring Panel
- Understanding How to Read the LDAP Connector Monitoring Panel

35.8.1 Displaying the Monitoring Panel

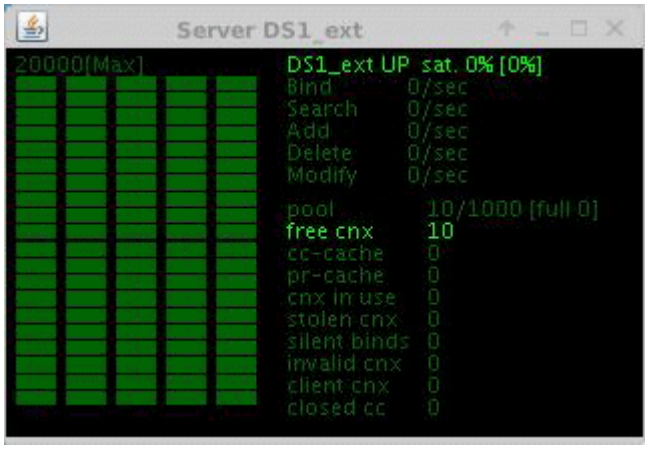
You must set the `MONITOR_LDAP_SERVER_EXTENSION` environment variable *before* starting the server to display the monitoring panel.

Run the `export` command as follows:

```
$ export MONITOR_LDAP_SERVER_EXTENSION=yes
$ start-ds
```

One monitoring panel displays information for one LDAP connector, similar to Figure 35-5.

Figure 35-5 Example LDAP Connector Monitoring Panel



35.8.2 Understanding How to Read the LDAP Connector Monitoring Panel

The LDAP Connector Monitoring Panel displays the throughput of each operation type, the status and saturation index of the server, the number of connections and so on.

You read the LDAP Connector Monitoring Panel's display as follows:

- The bar graph on the left indicates the throughput for each operation type, including Bind, Search, Add, Delete, and Modify operations.

> **Note:**
>
> Each bar-graph is limited to 20,000 operations/second, and each bar represents a throughput of 1000 operations/second. You can increase the limit from 20,000 to 100,000 by setting **`MONITOR_LDAP_SERVER_EXTENSION_MAX_THROUGHPUT=100000`** and restarting the server.

- The topmost entry on the right is the server status and indicates whether the remote LDAP server is UP or DOWN. For example, in Figure 35-5, server `ldap-01` is `UP`.
- The `Sat. 0% (0%)` field indicates the saturation index of the server.
  - A saturation index value of **`0%`** indicates that the server is fully operational.
  - A saturation index value of **`100%`** indicates that the server is saturated.
  - The value in parenthesis, **`(0%)`**, is the maximum value the saturation index has ever reached (peak value). Restarting the server resets the peak value to 0%.
- The remaining entries on the right indicate the current size of the connection pool and the number of connections. These entries include:
  - `pool x/y [full z]`

    Where
    - *x* is the current size of the connection pool (equal to the number of created connections)
    - *y* is the maximum pool size
    - *z* is the occurrence of "pool full" (always 0 in the current implementation)
  - `free cnx` is the number of free connections
  - `cc-cache` is the number of connections in the client-bound connection cache
  - `pr-cache` is the number of connections in the proxy-bound connection cache
  - `cnx in use` is the number of connections in use
  - `stolen cnx` is the number of connections being stolen in either cache
  - `silent binds` is the number of binds that the server is silently performing before using a connection.Oracle Unified Directory requests a silent bind on a connection when the connection is not bound yet or if the connection is bound with a non-relevant set of credentials.
  - `invalid cnx` is the number of connections being released as invalid
  - `client cnx` is the number of client connections that are currently connected and using the connector
  - `closed cc` is the number of closed client connections that have not yet been processed

  At any time, you should have the following invariant:

  ```
  pool = free-cnx + cnx in use + cc-cache + pr-cache
  ```

  If there is no on-going operation, then you should see the following count values set to 0 (any nonzero count reflects a connection management issue):

```
cnx in use = 0
client-cnx = 0
closed-cnx = 0
```

> **Note:**
>
> A *stolen connection* is a connection fetched from the `cc-cache` or `pr-cache` because there are no more free connections and the pool cannot extend anymore. A high number of stolen connections impacts the server performance because it implies many silent rebinds. To avoid stolen connections, increase the pool size.

35.9 Understanding the General Purpose Enterprise Monitoring Solutions

You can use a variety of general UNIX tools to monitor your server environment. For information about these tools, see the man pages on your UNIX system.

General purpose enterprise monitoring solutions is described in the following sections:

- About General UNIX Monitoring Tools
- About Solaris Monitoring Tools
- About HP-UX Monitoring Tools

35.9.1 About General UNIX Monitoring Tools

Review this topic for the general purpose UNIX monitoring tools can be used with Oracle Unified Directory.

The following table lists the general UNIX monitoring tools used with Oracle Unified Directory:

Tool	Description
`iostat`	Provides information about disk I/O and CPU usage.
`lsof`	Provides information about open file descriptors.
`lslk`	Provides information about file system locks.
`netstat`	Provides statistics about network functions.
`nslookup`	Allows you to query DNS servers for information about hosts and domains.
`ping`	Allows you to query the status of a remote host or network gateway.
`sar`	UNIX System V performance monitoring tool.
`tcpdump`	Allows you to debug and monitor network traffic.
`top`	Provides quick, easy monitoring of processes and CPU activities.
`trace`	Provides information about which system calls a process makes.
`traceroute`	Provides the path a packet takes throughout the Internet to reach its final destination.
`vmstat`	Provides statistics about process, virtual memory, disk, trap, and CPU activity.

35.9.2 About Solaris Monitoring Tools

Review this topic for the several Solaris monitoring tools can be used with Oracle Unified Directory.

The following table lists the Solaris monitoring tools used with Oracle Unified Directory:

Tool	Description
`lockstat`	Provides information about OS and application locking. Requires DTrace privileges.
`mpstat`	Provides statistics about each processor on the system.
`pmap`	Provides a breakdown of how much memory a process is using.
`proctool`	Monitors processes and threads.
`snoop`	Monitors network traffic. Indispensable when debugging low-level packets.
`SymbEL/Virtual\\Adrian`	Provides functionality of the above listed tools and more.
`truss`	Provides information about which system calls a process makes.

35.9.3 About HP-UX Monitoring Tools

Review this topic for the several HP-UX monitoring tools can be used with Oracle Unified Directory.

The following table lists the HP-UX monitoring tools used with Oracle Unified Directory:

Tool	Description
`glance`	Provides detailed system information about open file descriptors, locks, and threads.
`gpm`	GlancePlus is a graphical real-time performance diagnostic tool. Glance is the character-based component.
`tusc`	Provides a system call trapper.
`sysdef`	Provides information about kernel parameters.
`landiag`	Monitors network statistics.
`sam`	Provides a general system administration tool.

36
Tuning Performance

Oracle Unified Directory aims to be high-performing and highly-scalable. Though the server can achieve impressive results with the default server configuration and JVM settings, you can often significantly improve performance through some basic tuning.
The default settings of Oracle Unified Directory are targeted at evaluators and developers who run equipment with limited resources. When you deploy Oracle Unified Directory in a production environment, it useful to do some initial tuning of the Java Virtual Machine (JVM) and of the server configuration to improve scalability and performance (particularly for write operations).

Topics:

36.1 About Performance Problem Assessment

Examine the access log at `INSTANCE_DIR/OUD/logs/access` to get a quick idea of whether performance issues are related to problems with the server or with the client.

This log contains entries of the form:

```
[09/Sep/2009:15:36:18 +0200] SEARCH RES conn=1 op=16 msgID=17
  result=0 nentries=1 etime=1
```

The value of the `etime` field is the time (in milliseconds) that the server spent processing the request. Large `etimes` generally indicate an issue on the server side (which can usually be resolved by appropriate performance tuning or indexing). If you are experiencing performance problems but the `etimes` are small, the issue is more likely to be with your client application.

Comprehensive monitoring information is available under the `cn=monitor` entry. See Monitoring Oracle Unified Directory. You can also use Oracle Enterprise Manager to monitor Oracle Unified Directory performance. Refer to *Viewing Performance Metrics for an OUD Instance* in *Oracle® Enterprise Manager Plug-in forOracle Unified Directory User's Guide*.

36.2 Understanding How to Tune General Performance Parameters

Performance tuning strategies differ, depending on whether you run a directory server or a proxy server.

The following parameters can improve performance in specific deployment scenarios:

- **Java Version**. Use the most recent Java Runtime Environment (JRE) release available. Oracle Unified Directory works with Java 8 Update 131+.
- **Environment Variables**. The server uses the *OPENDS_JAVA_HOME* environment variable to point to your installed JRE. If you have multiple versions of Java installed on a system, set the *JAVA_HOME* environment variable to point to the root of the desired installation. In this way, the version of the JRE specified by the *JAVA_HOME* variable can be used by other applications but not by Oracle Unified Directory.

  To specify a JRE installation for the server, do one of the following:

  – Use the `dsjavaproperties` command to set the appropriate environment variables.

    For more information, see dsjavaproperties.

  – Set the *OPENDS_JAVA_BIN* environment variable (with the JAVA binary path).

  – Set the *OPENDS_JAVA_HOME* environment variable (with the JAVA installation path).

36.3 Understanding Java Virtual Machine Settings Using `dsjavaproperties` Utility

You can use the `JAVA_ARGS` environment variable to provide global configuration arguments that can be passed to the JVM, or you can use the `java.properties` file. Any argument that can be used with the `java` command can be used with both methods.

It is recommended to tune the JVM for optimal performance and ensures that Oracle Unified Directory applications are robust and responsive. You can tune the JVM by tuning the heap size. The heap size is divided into the following:

- Young generation: Includes operations like PDUs and local variables.
- Old generation: Includes Oracle Unified Directory caches like the JE database cache and the entry cache.
- Permanent generation: Includes constants and classes.

When Oracle Unified Directory is in Directory Server mode, you can perform one of the following database caching options:

- Cache the entire database in database cache. This will give optimal performance but will lead to long cache warmup and larger heap size.
- Cache only the internal nodes of the database Btree (Upper and inner nodes) in database cache and keep remaining RAM for file system cache. This will give good performance, short cache warmup, smaller heap size and is recommended for very large deployments (Above 50MBytes entries). It is recommended for small and medium deployments.

For more information, see Determining the Database Cache Size.

> **Note:**
>
> For proxy mode, use large old generation for distribution with global index.

For more information, see dsjavaproperties.

For additional information about tuning the JVM, see the Java Performance Documentation (`http://java.sun.com/docs/performance/`).

The *Java Tuning White Paper* and *Garbage Collection Tuning* documents, both at http://www.oracle.com/technetwork/java/performance-138178.html, are also particularly useful.

The following table describes the main JVM tunable options:

Parameter	Description
`-server`	Always use the server JVM instead of the client JVM. The client VM is better optimized for processes that run for a short period of time and need to start as quickly as possible. The server VM can take longer to warm up but is faster in the long run.
`-XX:+UseCompressedOops`	Use this option if you use the 64-bit JVM and if the heap size is less than 32 Gbytes.
`-Xms2g` and `-Xmx2g`	This parameter sets the initial and maximum heap size available to the JVM. Increasing the heap size can improve performance, but setting it too high can have a detrimental effect in the form of longer pauses for full garbage collection runs. The initial and maximum sizes should generally be set to the same values. For maximum performance, size the heap so that the entire DB can be cached in memory. In general, you should allocate enough heap for the server runtime and the rest to the DB cache. For example, if you want to modify the heap size of an Oracle Unified Directory instance with only one JE back end named `userRoot`. Then you must decide the space needed for the new generation, the old generation and the perm generation. To size the different generations, you must consider the following: • The size of the database impacting the old generation • Determine the need to use an entry cache impacting the old generation. • The type of GC used impacting the old generation. • The type of usage impacting the new generation. If you use CMS as the garbage collector of the oldgen, then you must consider the `-XX:CMSInitiatingOccupancyFraction` property when calculating the heap size so that it is coherent with the size (or percent of the heap) occupied by the dbcache. If you set the `CMSInitiatingOccupancyFraction` to 55, then set the dbcache percent to 50. Then, if you have a database on disk that is 10GB, you need at least a heap of 22GB for the entire database to fit into the dbcache.
`-XX:+UseG1GC`	Use the G1GC garbage collector to minimize the response time of LDAP operations while maintaining balanced throughput and predictable pause times. G1GC divides the heap into regions and prioritizes collecting the regions with the most garbage.
`-XX:LargePageSizeInBytes=256m`	Use large pages for the information it stores in memory. This argument applies primarily to systems using the UltraSPARC T1 processor.
`-XX:+UseParallelGC`	Specify that the system should use parallel garbage collection, which is particularly useful on systems with a large number of CPUs.
`-XX:ParallelGCThreads=8`	Specify that the JVM should use 8 threads when performing parallel garbage collection. By default, the number of threads equals the number of CPUs, but this can be inappropriate on systems with a very large number of CPUs or on CMT-based systems like those using the UltraSPARC T1 processor.

36.4 Tuning Java Virtual Machine Settings Using the `dstune` Utility

The `dstune` command-line utility allows you to tune the Oracle Unified Directory server and tools (`import-ldif`, `export-ldif`, `rebuild-index`, and `verify-index`) using criteria such as the data that the directory contains or the amount of system memory to use.

Any changes made using the `dstune` utility take effect when the server is restarted.

This section includes the following topics:

- Using the `dstune` Utility
- Executing the Interactive Mode of the `dstune` Utility

> **Note:**
>
> The various tuning options, described in Understanding the Tuning Options Provided by the `dstune` Utility, are available only if you run a Java Virtual Machine that uses Java HotSpot, such as the Oracle Java Standard Edition. If you run a JVM without Java Hotspot, the `dstune` memory-based and data-based options are not available.

36.4.1 Using the `dstune` Utility

The `dstune` utility has several tuning options. You can tune the Oracle Unified Directory server based on memory limits or LDAP data information or by providing runtime options.

The following topics describe the different tuning options available using the `dstune` utility:

- Understanding the Tuning Options Provided by the `dstune` Utility
- Displaying the Current Tuning Mode

36.4.1.1 Understanding the Tuning Options Provided by the `dstune` Utility

> **Note:**
>
> Beginning with Oracle Unified Directory11*g* Release 2 (11.1.2.3), the `dstune automatic` subcommand is no longer available (but `automatic` usage is still available for backward compatibility).
>
> To specify automatic tuning similar to previous versions of Oracle Unified Directory, use the `dstune set-runtime-options` subcommand with the `--value autotune` suboption. See About Runtime Tuning.
>
> For more information about the `dstune` subcommands and options, see dstune

The `dstune` utility allows you to tune the server and tools based on the following tuning options:

- About Data-Based Tuning
- About Memory-Based Tuning

- About Runtime Tuning

36.4.1.1.1 About Data-Based Tuning

The data-based tuning mode (`dstune data-based` subcommand) allows you to tune the Oracle Unified Directory server based on the data that the contents of the database will contain or currently contains.

To provide information about the data that the database will contain, specify the number of entries (`--entryNumber` suboption) and the average size in kilobytes of the entries (`--entrySize` suboption).

You can also specify a path to an LDIF file (`-l` or `--ldifFile` suboption) that contains the data to tune the server.

If you do not provide options or the information about the data that the database will contain, the `data-based` subcommand analyzes the contents of the current database and determines the recommended minimum and optimal memory values for that data.

In non-interactive mode, `dstune` uses a default memory value to tune the server (and displays the memory value used).

In interactive mode, `dstune` asks you for a memory value, but it also presents some recommendations.

36.4.1.1.2 About Memory-Based Tuning

The memory-based tuning mode (`dstune mem-based` subcommand) allows you to tune the Oracle Unified Directory server and tools based on the heap size they will use.

To specify the memory to be used for the server or tools, use the *`--memory heap-size`* suboption.

If you are tuning the server, you can specify the system memory (`systemMemory` option) as either an amount or percentage:

- Amount: For example, to use 2 GB, specify `systemMemory:2g`. The `dstune` utility then splits the value you provide into two parts: the heap size that the Java Virtual Machine of the server will use and an estimation of the required file-system cache.
- Percentage: For example, to dedicate 50 percent of system memory to the server, specify `systemMemory:50.0%`. To fully dedicate a machine to the server, specify `systemMemory:100%`.

By default, `dstune` tunes the server, but the `--targetTool` option allows you to specify the other tools to tune.

36.4.1.1.3 About Runtime Tuning

The runtime tuning mode (`dstune set-runtime-options` subcommand) allows you to use the JVM default values on the system or to directly provide JVM arguments to tune the Oracle Unified Directory server and tools.

The automatic tuning mode (`--value autotune` suboption) also allows you to tune the server and each tool automatically each time they are launched.

By default, `dstune` tunes the server, but the `--targetTool` option allows you to specify the tools you want to tune.

36.4.1.2 Displaying the Current Tuning Mode

You use the `dstune list` subcommand to display the current tuning settings of an Oracle Unified Directory server and the tools.

For example, the following command displays the current tuning settings of an Oracle Directory Server instance and tools.

```
$ dstune list
Tool : Tuning Value
-------------:-------------------------------------------------------------------
server        : -Xms9062m -Xmx9062m -XX:+UseCompressedOops -server -Xmn1g
              : -XX:MaxTenuringThreshold=1 -XX:+UseG1GC -XX:-G1UseAdaptiveIHOP
              : -XX:InitiatingHeapOccupancyPercent=60
import-ldif   : -Xms9062m -Xmx9062m -XX:+UseCompressedOops -server -Xmn1g
              : -XX:+UseParallelGC -XX:+UseNUMA
export-ldif   : -Xms9062m -Xmx9062m -XX:+UseCompressedOops -server -Xmn1g
              : -XX:+UseParallelGC -XX:+UseNUMA
rebuild-index : -Xms9062m -Xmx9062m -XX:+UseCompressedOops -server -Xmn1g
              : -XX:+UseParallelGC -XX:+UseNUMA
verify-index  : -Xms9062m -Xmx9062m -XX:+UseCompressedOops -server -Xmn1g
              : -XX:+UseParallelGC -XX:+UseNUMA
```

36.4.2 Executing the Interactive Mode of the `dstune` Utility

The interactive mode of the dstune utility provides the capability to tune Oracle Unified Directory and tools server based on memory limits or LDAP data information or by providing runtime options.

- Setting Memory-Based Tuning Options
- Setting Data-Based Tuning Options
- Setting Runtime Tuning Options
- Displaying the Current Tuning Settings

> **Note:**
>
> For more information about the tuning options, see Understanding the Tuning Options Provided by the `dstune` Utility.

36.4.2.1 Setting Memory-Based Tuning Options

You use the interactive mode of the `dstune` utility to tune Oracle Unified Directory server using memory-based tuning options.

This section contains the following topics:

- Tuning Oracle Unified Directory Server by Specifying a Heap Size
- Tuning Oracle Unified Directory Server by Specifying a Percentage of System Memory

36.4.2.1.1 Tuning Oracle Unified Directory Server by Specifying a Heap Size

The example in this section describes how to run the `dstune` utility in interactive mode to tune Oracle Unified Directory server by specifying 2 Gbytes for the heap size.

Run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 1

You must provide the tools the runtime options will apply to.
If you want to use the settings for the server, provide the value 'server'.
If you want to use the settings for all the tools, provide the value 'all'.
The other allowed values are import-ldif, export-ldif, rebuild-index,
verify-index.
You can provide several values separated with a comma (for instance
'export-ldif,rebuild-index').

Tools [server]:

You have chosen to tune the server.  To tune the server you can provide
directly the Java heap size to be used by the server, you can specify the
amount of system memory to be used providing a percentage (use 100 % to
dedicate the machine to the OUD server) or you can specify the amount of
system memory (the sum of the Java Heap and an estimation of the required File
System Cache).

    1)  Provide the heap size
    2)  Provide the percentage of system memory to be used by the server
    3)  Provide the size of system memory to be used by the server

Enter choice [1]:

You must provide the heap size to be used by the tools.
To specify a value in megabytes, use 'm' after the value (for instance 768m).
For gigabytes, use 'g' (for instance 2.5g).
If no unit is specified after the value, megabytes will be used.

Heap Size [2.27g]: 2g

Calculating Tuning Settings ..... Done.
Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

36.4.2.1.2 Tuning Oracle Unified Directory Server by Specifying a Percentage of System Memory

The example in this section describes how to run the `dstune` utility in interactive mode to tune Oracle Unified Directory server by specifying 50 percent of system memory to be used by the server. The server has 100,000 entries.

Run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 1

You must provide the tools the runtime options will apply to.
If you want to use the settings for the server, provide the value 'server'.
If you want to use the settings for all the tools, provide the value 'all'.
The other allowed values are import-ldif, export-ldif, rebuild-index,
verify-index.
You can provide several values separated with a comma (for instance
'export-ldif,rebuild-index').

Tools [server]:

You have chosen to tune the server.  To tune the server you can provide
directly the Java heap size to be used by the server, you can specify the
amount of system memory to be used providing a percentage (use 100 % to
dedicate the machine to the OUD server) or you can specify the amount of
system memory (the sum of the Java Heap and an estimation of the required File
System Cache).

    1)  Provide the heap size
    2)  Provide the percentage of system memory to be used by the server
    3)  Provide the size of system memory to be used by the server

Enter choice [1]: 2

To be able to properly calculate the Java heap size for a given percentage,
the contents of the database will be analyzed.

Reading the Server Configuration ..... Done.
Reading the Database Contents ..... Done.

The memory you assign for OUD will be divided in two parts: the Java heap size
of the OUD process and the estimated memory that will be required for the file
system cache.
Provide the percentage of the memory that should be assigned to OUD (use 100%
if you want to have a dedicated machine for this OUD server).

Memory Percentage [25.73]: 50

The specified percentage corresponds to the following memory values:
2.83 GB: 1.59 GB (OUD Java Heap Size) + 1.25 GB (Estimated File System Cache)
Do you want to use this value? (yes / no) [yes]:

Calculating Tuning Settings ..... Done.
Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

36.4.2.2 Setting Data-Based Tuning Options

You use the interactive mode of the `dstune` utility to tune Oracle Unified Directory server using database-based tuning.

This section contains the following topics:

- Tuning Oracle Unified Directory Server Using the Contents of the Database
- Tuning Oracle Unified Directory Server by Providing an LDIF File

36.4.2.2.1 Tuning Oracle Unified Directory Server Using the Contents of the Database

The example in this section describes how to run the `dstune` utility in interactive mode to tune Oracle Unified Directory server using the data that the server currently contains (that is, the current contents of the database).

Run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 2
Provide information about the LDAP data that will be used to tune the server.
You can choose to tune the server based on its current contents, you can
provide an LDIF File with the data, or directly the number and average size of
your entries.

    1)  Use the data that the server contains currently
    2)  Use the contents of an LDIF file
    3)  Use the number of entries

    c)  cancel

Enter choice [1]:

To calculate the tuning options, the contents of the database will be
analyzed.

Reading the Server Configuration ..... Done.
Reading the Database Contents ..... Done.

Memory Requirements Information for the Data in the Server:

System Memory:                   5.66 GB
Recommended Min. Memory:         444.76 MB (7.67 % of System Memory)
                                 288.73 MB (Java Heap) +  156.04 MB (Estimated
                                 File System Cache)
Memory for Optimal Performance:  1.46 GB (25.73 % of System Memory)
                                 1.30 GB (Java Heap) +  0.15 GB (Estimated File
                                 System Cache)
================================================================================
Recommended Memory:              1.46 GB (25.73 % of System Memory)
                                 1.30 GB (Java Heap) +  0.15 GB (Estimated File
                                 System Cache)

You must provide the memory you want to use.  You can provide the Java heap,
the amount of system memory or the percentage of the system memory that you
want the OUD server to use.
```

```
    1)  Provide the heap size
    2)  Provide the percentage of system memory to be used by the server
    3)  Provide the size of system memory to be used by the server

Enter choice [1]:

You must provide the heap size to be used by the tools.
To specify a value in megabytes, use 'm' after the value (for instance 768m).
For gigabytes, use 'g' (for instance 2.5g).
If no unit is specified after the value, megabytes will be used.

Heap Size [1.30g]:

Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

36.4.2.2.2 Tuning Oracle Unified Directory Server by Providing an LDIF File

The example in this section describes how to run the `dstune` utility in interactive mode to tune Oracle Unified Directory server by providing an LDIF file with 200,000 entries.

Run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 2
Provide information about the LDAP data that will be used to tune the server.
You can choose to tune the server based on its current contents, you can
provide an LDIF File with the data, or directly the number and average size of
your entries.

    1)  Use the data that the server contains currently
    2)  Use the contents of an LDIF file
    3)  Use the number of entries

    c)  cancel

Enter choice [1]: 2

LDIF File Path: /tmp/example.ldif
Calculating tuning settings based on the contents of the LDIF file ..... Done.

Analyzing file /scratch/joverga/servers/example.ldif (around 2 seconds
remaining) ..... Done.

Memory Requirements Information for the LDIF File:

System Memory:                   5.66 GB
Recommended Min. Memory:         550.79 MB (9.50 % of System Memory)
                                 321.46 MB (Java Heap) +  229.33 MB (Estimated
                                 File System Cache)
Memory for Optimal Performance:  1.67 GB (29.52 % of System Memory)
                                 1.45 GB (Java Heap) +  0.22 GB (Estimated File
                                 System Cache)
```

```
================================================================================
Recommended Memory:              1.67 GB (29.52 % of System Memory)
                                 1.45 GB (Java Heap) +  0.22 GB (Estimated File
                                 System Cache)

You must provide the memory you want to use.  You can provide the Java heap,
the amount of system memory or the percentage of the system memory that you
want the OUD server to use.

    1)  Provide the heap size
    2)  Provide the percentage of system memory to be used by the server
    3)  Provide the size of system memory to be used by the server

Enter choice [1]:

You must provide the heap size to be used by the tools.
To specify a value in megabytes, use 'm' after the value (for instance 768m).
For gigabytes, use 'g' (for instance 2.5g).
If no unit is specified after the value, megabytes will be used.

Heap Size [1.45g]:

Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

36.4.2.3 Setting Runtime Tuning Options

You use the interactive mode of the `dstune` utility to set the runtime tuning options for the server and tools.

The example in this section sets the JVM options for the server and the `export-ldif` tool.

To tune the Oracle Unified Directory Server using the JVM Options, run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 3

You must provide the tools the runtime options will apply to.
If you want to use the settings for the server, provide the value 'server'.
If you want to use the settings for all the tools, provide the value 'all'.
The other allowed values are import-ldif, export-ldif, rebuild-index,
verify-index.
You can provide several values separated with a comma (for instance
'export-ldif,rebuild-index').

Tools [server]: server,export-ldif

You must provide the runtime options you want to use.
If you want to use automatic tune, provide the value 'autotune'.
If you want to use the default settings of the Java Virtual Machine on your
system, provide the value 'jvm-default'.
You can also provide directly the Java arguments that the tools must use.
```

```
Runtime Options [autotune]: -server -Xmx2048m

Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

36.4.2.4 Displaying the Current Tuning Settings

The example in this section runs the `dstune` utility in interactive mode to display the current tuning settings of an Oracle Unified Directory server and tools.

Run the `dstune` utility as follows:

```
$ dstune
What do you want to do?

    1)  Tune based on memory limits
    2)  Tune based on LDAP data information
    3)  Tune providing runtime options
    4)  List the current tuning settings

    q)  quit

Enter choice: 4

Tool          : Tuning Value
--------------:-------------------------------------------------------------------
server        : -Xms853m -Xmx853m -d32 -server -XX:MaxTenuringThreshold=1
              : -XX:+UseConcMarkSweepGC -XX:CMSInitiatingOccupancyFraction=55
import-ldif   : -Xms715m -Xmx715m -d32 -server -XX:+UseParallelGC -XX:+UseNUMA
export-ldif   : -Xms715m -Xmx715m -d32 -server -XX:+UseParallelGC -XX:+UseNUMA
rebuild-index : -Xms715m -Xmx715m -d32 -server -XX:+UseParallelGC -XX:+UseNUMA
verify-index  : -Xms715m -Xmx715m -d32 -server -XX:+UseParallelGC -XX:+UseNUMA
```

36.5 Determining the Database Cache Size

If you have installed or configured and initialized an Oracle Unified Directory instance then you can determine the database cache size requirements by measuring the size of `<OUD_INSTANCE_DIR>/OUD/db/userRoot` directory (Assuming there is only one database for the Oracle Unified Directory instance named `userRoot`).

If an Oracle Unified Directory instance is not configured or initialized, then you can determine the memory required to store internal nodes for one index file or the file containing user data, by running the DbCacheSize utility (`com.sleepycat.je.util`).

For more information on using the DbCacheSize utility, see this Javadoc page: http://docs.oracle.com/cd/E17277_02/html/java/com/sleepycat/je/util/DbCacheSize.html.

For example, 10 million entries of 4Kbytes with an index and average key size of 10 bytes are as follows:

```
[oud@oel5 bin]$ java -jar -XX:+UseCompressedOops /space/Middleware/
Oracle_OUD1/lib/je.jar DbCacheSize -records 10000000 -key 10 -data 4000

=== Database Cache Size ===
 Minimum Bytes    Maximum Bytes   Description
---------------  ---------------  -----------
    259,725,752      317,907,896  Internal nodes only
 40,721,011,192   40,779,193,336  Internal nodes and leaf nodes
```

```
=== Internal Node Usage by Btree Level ===
 Minimum Bytes    Maximum Bytes      Nodes    Level
---------------  ---------------  ----------  -----
    256,180,800      313,709,120     112,360    1
      3,503,312        4,149,456       1,262    2
         38,864           46,032          14    3
          2,776            3,288           1    4
```

A 10 million entries deployment with 4 Kbytes will require 37 Gbytes to store the full user data in the database cache (4Kbytes entries and the internal nodes of the Database Btree). If you want to store only the internal nodes in the database cache, then 303 Mbytes are required per indexes (3 Gbytes for 10 indexes).

36.6 Tuning the Server Configuration

Various components of the server can be tuned to provide performance improvements in specific scenarios. Most performance tuning recommendations depend on several variables, including the anticipated workload, the types of data that are stored, and the hardware and resources available.

The following topics provide some general tuning recommendations for performance improvement in specific deployments:

- Back End Tuning Parameters
- Core Server Tuning Parameters
- Tuning a Server Containing Static Groups
- Additional Tuning Recommendations

36.6.1 Back End Tuning Parameters

Review this topic for the different Berkeley DB JE parameters to tune performance improvements of the Berkeley Database.

The following table lists these parameters to tune performance:

Parameter	Description
`je.checkpointer.highPriority`	If true, the checkpointer uses more resources to complete the checkpoint in a shorter time interval. Btree latches are held and other threads are blocked for a longer period. Log cleaner record migration is performed by cleaner threads instead of lazily during eviction and checkpoints (see CLEANER_LAZY_MIGRATION). When set to true, application response time may be longer during a checkpoint, and more cleaner threads may be required to maintain the configured log utilization. Setting that property to false is a way to achieve better throughput and lower response times.
`preload-time-limit`	You can configure the server to preload some database contents into memory on startup. For large databases, preloading the database cache avoids a long *warmup* period after server startup. For more information, see "Local DB Backend Configuration" in the *Configuration Reference for Oracle Unified Directory*.

Parameter	Description

db-cache-percent and db-cache-size	Use these properties to configure the amount of memory that the database cache uses. For best performance, consider configuring the server so that the whole database fits into the database cache.
	Determine the approximate size of the database after an import. For example, after doing an import into the userRoot back end, run the following command (on UNIX systems) to determine the size of the database:
	`$ cd INSTANCE_DIR/OUD/db`
	`$ du -sk userRoot/`
	`910616 userRoot/`
	On Windows systems, use an equivalent procedure to determine the database size. Remember that the database size is not static and can increase after an initial import when modifications are made.
	Setting the JVM heap to 2 Gbytes (-Xms2g -Xmx2g), and the db-cache-percent to 50, will cause the DB cache to use 1 Gbyte of memory. To monitor the DB cache size, observe the following properties under the "dn:cn=userRoot Database Environment,cn=monitor" entry through Jtrace and JMX:
	• Check that EnvironmentCacheDataBytes has a value that is consistent with the expected size of the DB cache.
	• Check that EnvironmentNCacheMiss does not have unexpected growth when loading the server.
	As the database grows very large over time due to replication metadata, users, and applications. This may effect the performance after the import. It is recommended that you tune the Oracle Unified Directory JVM heap size (Primarily the old generation).

db-directory	Ensure that the database is held on a fast file system with adequate storage. The file system should be different to the location of the access logs. By default, the database will grow to twice its original size. For example, if the database is 1 Gbyte after an import, the file system should have at least 2 Gbytes available.
db-evictor-lru-only	Use this property can be used to control how the database cache retains information. Setting this value to false ensures that the internal nodes are maintained in cache, which provides better performance when the JE cache holds only a small percentage of the database contents.
db-txn-durability	Use this property to configure durability for write operations. Reducing durability can increase write performance, but it can also increase the chance of data loss if your JVM or system crashes. This property takes the following values: • **write-to-disk.** All data are written synchronously to disk. • **write-to-filesystem.** Data are written to the file system immediately, but might stay in the file system before being flushed to disk. • **write-to-cache.** Data are written to an internal buffer and flushed to the file system, then to disk when necessary.
db-log-file-max	Use this property to control the size of JE log files. Increasing the file size can improve write performance, but it can also make it harder to maintain the desired utilization percentage.
db-num-cleaner-threads and db-cleaner-min-utilization	These properties control how the cleaner works, which keeps the database size down and keeps up with high write throughput.
db-num-lock-tables	On systems with a large number of CPUs, this property can improve concurrency within the database lock manager.
tombstone-purge-interval	Specifies the time interval after which tombstone purging restarts.
db-evictor-nodes-per-scan	This maps to the je.evictor.nodesPerScan property of the JEB Environment Configuration.

Parameter	Description
db-checkpointer-bytes-interval	Specifies the maximum number of bytes that may be written to the database before it is forced to perform a checkpoint. This maps to the je.checkpointer.bytesInterval property of the JEB Environment Configuration.
db-logging-level	Specifies the log level used by the database when it is writing information into the je.info file. The database trace logging level (in increasing order of verbosity) is chosen from: OFF, SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST, ALL.
disk-low-threshold and disk-full-threshold	Low disk threshold to limit database updates. Specifies the low free space on the disk. Full disk threshold to limit database updates. These parameters have to be in proportion to db-log-file-max.
je-property: je.cleaner.lookAheadCacheSize	The look ahead cache size for cleaning in bytes. Increasing this value can reduce the number of Btree lookups. This maps to the je.cleaner.lookAheadCacheSize property of the JEB Environment Configuration.
je-property: je.cleaner.bytesInterval	The cleaner checks disk utilization every time you write this many number of bytes to the log. When overriding the default value, you must use caution to ensure that the cleaner is woken frequently, so that the reserved files are deleted quickly to avoid violating a disk limit. This maps to the je.cleaner.bytesInterval property of the JEB Environment Configuration.

The following changes are applicable for the default data backend (userRoot) for new OUD instances only. Older defaults will apply for existing instances unless the configuration has been overridden (which is persisted in config.ldif).

- db-evictor-lru-only: false
- tombstone-purge-interval: 30 min
- db-evictor-nodes-per-scan: 100
- db-checkpointer-bytes-interval: 500 MB
- db-cleaner-min-utilization: 75
- disk-low-threshold: 200 MB
- disk-full-threshold: 100 MB
- je-property: je.cleaner.lookAheadCacheSize: 1 MB
- je-property: je.cleaner.bytesInterval: 100 MB

The following changes are applicable to new instances and existing instances, unless configuration has been overridden:

- db-logging-level: The default value is set to WARNING for all backends in general and also for the default data backend (userRoot) in specific.
- db-num-cleaner-threads: It is set to 10 for the default data backend. The default value of this property for all other backends is a maximum of 10 or number of CPUs, unless configuration has been overridden (persisted in config.ldif). You must bear in mind that higher number of cleaner threads affects OUD performance, because it needs to obtain frequent write locks.

36.6.2 Core Server Tuning Parameters

Review this topic for the different parameters to tune performance improvements of the core server.

The following table lists the following core server tuning parameters to tune performance:

Parameter	Description
num-request-handlers	You can configure this property to enable the LDAP connection handler (and the LDAPS connection handler, if it is enabled) to use multiple threads for decoding client requests. Increasing the number of threads on systems with a larger number of CPUs can improve performance. As a rule of thumb, set this property to a quarter the number of CPUs, with a maximum of twelve. In some cases, disabling the `keep-stats` property can help reduce lock contention in the connection handlers. For more information, see "LDAP Connection Handler Configuration" in the *Configuration Reference for Oracle Unified Directory*.
num-worker-threads	The default value of this property is two times the number of CPUs or 100, whichever is higher. The default value of this property is ten times the number of CPUs or 100, whichever is higher.
log-file	Ensure that the access log publisher is on a fast file system, or turn it off altogether by setting the `enabled` property to `false`. For more information see "File Based Access Log Publisher Configuration" in the *Configuration Reference for Oracle Unified Directory*.

36.6.3 Tuning a Server Containing Static Groups

Review these topics for some recommendations on performance improvement if your server contains static groups.

- Enabling a FIFO Group Entry Cache
- Configuring FIFO Group Entry Cache Properties
- Evaluating **`Member or Uniquemember`** Indexes
- Managing Static Groups With More Than 100,000 Members
- Importing Static Groups

36.6.3.1 Enabling a FIFO Group Entry Cache

You can improve the overall performance of the server by storing static groups in a FIFO Group Entry Cache. Storing static groups in this cache reduces the time required to perform group membership lookup, which is necessary in evaluating ACIs for example.

> **Note:**
>
> For more information, see "FIFO Group Entry Cache Configuration" in the *Configuration Reference for Oracle Unified Directory*.

For example, to create a new FIFO Group Entry cache using `dsconfig`, run the following command:

```
$ ./dsconfig create-entry-cache -t fifo-group --cache-name myGroupCache \
--set cache-level:1 --set enabled:true -n
```

To list properties of this FIFO Group Entry cache using `dsconfig`, run the following command:

```
$ ./dsconfig get-entry-cache-prop --cache-name myGroupCache
Property           : Value(s)
-------------------:---------------------------------------------------------
cache-level        : 1
enabled            : true
exclude-filter     : (objectClass=ds-virtual-static-group)
include-filter     : (|(objectClass=groupOfNames)(objectClass=groupOfUniqueName
                   : s)(objectClass=groupOfEntries))
max-entries        : 2147483647
max-memory-percent : 5
```

36.6.3.2 Configuring FIFO Group Entry Cache Properties

Oracle Unified Directory supports several kinds of cache to enhance the performance especially of the database cache and the file system cache. FIFO Group Entry Caches use a FIFO queue to keep track of the cached entries. They are designed to cache large static group entries.

In deployment scenarios involving large static groups, you can configure the entry cache to include the group entries to accelerate group membership and group-based ACI evaluation. To do so, you can configure the following properties:

- include-filter: Specifies a set of filters to define criteria for determining which entries should reside in the entry cache. If a filter list is provided, then only entries matching at least one of the given filters is stored in the cache.

  For instance, entries matching the following LDAP criteria will be stored in the entry cache:

  ```
  "(|(objectClass=groupOfNames)(objectClass=groupOfUniqueNames)
  (objectClass=groupOfEntries))"
  ```

  You can also split the preceding filter as follows:

  Default:

  ```
  (|(objectClass=groupOfNames)(objectClass=groupOfUniqueNames)
  (objectClass=groupOfEntries))
  ```

  After splitting:

  ```
  ds-cfg-include-filter: (objectClass=groupOfNames)
  ds-cfg-include-filter: (objectClass=groupOfUniqueNames)
  ds-cfg-include-filter: (objectClass=groupOfEntries)
  ```

  The Default and the After behavior is the same. In both the scenarios, if any one of the filter matches, then it adds that entry in the group cache.

  It is worth mentioning that if `ds-cfg-include-filter` property is not provided, then it will return the all the static groups.

- cache-level: Specifies the cache level in the cache order in which the cache will be configured or processed if more than one instance of the cache is configured.

  By default, the `cache-level` is set to `1` for FIFO Group Cache. However, if you want to define a different cache type, for instance Soft Reference or File System Entry cache then you must provide a different value for `cache-level` property. In other words, the `cache-level` should not be the same as defined for FIFO Group Cache as `1`. You must define the cache level as more than one.

  As part of server tuning for performance improvement, FIFO User Cache is enabled by default with a `cache-level` set to `1`. So by default, the `cache-level` is set to `2` for FIFO Group Cache. This applies to new OUD instances only.

36.6.3.3 Evaluating `Member or Uniquemember` Indexes

Evaluation of group membership (`ismemberof` attribute, ACIs) is using the `member` and `uniquemember` indexes. To improve server performance, it is recommended that you set the `index-entry-limit` property for these indexes to a value that is greater than the maximum number of groups to which a user can belong.

> **Note:**
>
> For more information, see "Local DB Index" in the *Configuration Reference for Oracle Unified Directory*.

For example, to set the `index-entry-limit` property of the `member` index to 5,000 using `dsconfig`, run the following command:

```
$ ./dsconfig set-local-db-index-prop --element-name userRoot --index-name member --set
index-entry-limit:5000
```

After setting the `index-entry-limit` property, it is recommended that you rebuild the index. For example,

```
$ rebuild-index -b dc=example,dc=com -i member
```

> **Note:**
>
> For more information, see Maintaining Indexes.

To list properties of the member index using `dsconfig`, run the following command:

```
$ ./dsconfig get-local-db-index-prop --element-name userRoot --index-name member
Property                      : Value(s)
------------------------------:---------
attribute                     : member
index-entry-limit             : 5000
index-extensible-matching-rule : -
index-type                    : equality
```

36.6.3.4 Managing Static Groups With More Than 100,000 Members

You can manage static groups by defining some limits for the following operations:

- `MOD`, `MODDN`, and `DEL` operations on static groups:

  If the operation exits with an `administrative limit exceeded` error, then you can increase the server's `member-lookthrough-limit` property value.

- `SEARCH` operations on groups:

  An `ASN.1` error occurs when a `SEARCH` operation returns a static group entry containing more than 100,000 members. You can increase the maximum number of returned members by changing the server's `returned-attribute-value-limit` property.

When increasing these various limits, you must tune the allocated memory for the Java Virtual Machine accordingly.

> **Note:**
>
> For more information about the preceding operations, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

Following are some examples:

- To set the `member-lookthrough-limit` property to 200,000 using `dsconfig`, run the following command:

```
$ ./dsconfig set-global-configuration-prop --advanced --set member-lookthrough-
limit:200000
```

- To set the `returned-attribute-value-limit` property to 150,000 using `dsconfig`, run the following command:

```
$ ./dsconfig set-global-configuration-prop --advanced --set returned-attribute-value-
limit:150000
```

- To retrieve the value of these properties using `dsconfig`, run the following command:

```
 $ ./dsconfig get-global-configuration-prop --advanced
--property returned-attribute-value-limit --property member-lookthrough-limit
 Property                       : Value(s)
 -------------------------------:---------
 member-lookthrough-limit       : 200000
 returned-attribute-value-limit : 150000
```

36.6.3.5 Importing Static Groups

Oracle Unified Directory has introduced a new server-wide configuration parameter, `import-big-entries-memory-percent`, to allow importing big entries like big group entries.

The following topics describe how to import static groups:

- Overview of Importing Static Groups
- Setting the Configuration Parameter for Importing Static Groups

36.6.3.5.1 Overview of Importing Static Groups

Whether you perform the import online or offline, `import-big-entries-memory-percent` represents the amount of memory that the JVM will allocate to allow big entries, such as big group entries to be loaded into memory after they have been read from the imported LDIF file, so that they fit into memory.

- For offline imports, the JVM performing the import is the `import-ldif` command JVM.
- For online imports,Oracle Unified Directory creates an administrative import task in the JVM of the server and the import is performed inside the server's JVM.

When the import is launched, a certain amount of memory is available in the considered process. (Some memory has already been reserved for other components, such as the DB cache.) The `import-big-entries-memory-percent` represents what percentage of this free memory will be reserved to enable big entries from the LDIF file to load. The import machinery uses the rest of the memory.

The default `import-big-entries-memory-percent` value is 10%. If you import LDIF files with big group entries (for example, millions of members), and if the JVM is running out of memory, it would be worse if you increased the percentage value to something like 80-90%.

If you run out of memory when importing big groups entries, then you should tune the `import-big-entries-memory-percent` parameter, but you should also consider increasing the JVM heap size, tuning `java.properties`, and running the `dsjavaproperties` command afterward.

> **Note:**
>
> For more information, see "Global Configuration" in the *Configuration Reference for Oracle Unified Directory*.

36.6.3.5.2 Setting the Configuration Parameter for Importing Static Groups

You use the `import-big-entries-memory-percent` property for setting the configuration parameter for large static groups.

For example, to set the `import-big-entries-memory-percent` property to 20% using `dsconfig`, run the following command:

```
$ ./dsconfig set-global-configuration-prop --set import-big-entries-memory-percent:20
```

To retrieve the value of the `import-big-entries-memory-percent` property using `dsconfig`, run the following command:

```
$ ./dsconfig get-global-configuration-prop --property import-big-entries-memory-percent
 Property                          : Value(s)
 ----------------------------------:---------
 import-big-entries-memory-percent : 20
```

36.6.4 Additional Tuning Recommendations

You can further improve the server performance in specific scenarios by referring these recommendations.

Note the following:

- **Enable an Entry Cache**. In some cases, particularly those involving relatively small directories (for example, up to a few hundred thousand entries), it can be useful to enable an entry cache. In general the FIFO entry cache provides better results than the soft reference entry cache. For more information, see "Entry Cache Configuration" in the *Configuration Reference for Oracle Unified Directory.*

For large database, it is recommended that you store only a specific set of the data in the cache, by using the `include-filter` property.

FIFO User Entry Cache is enabled by default with `cache-level` set to `1`. FIFO Group Entry Cache is set to `2`. This applies only to new instances. Existing instances are not impacted and they continue to use the FIFO cache as configured.

The FIFO User Entry Cache is configured to include entries only from (`objectClass=Person`) using the `include-filter`.

- **Disable Unused Virtual Attributes**. If the functionality needed by one or more of the virtual attributes is not required, they can be disabled for a slight performance improvement when decoding entries. For more information, see "Virtual Attribute Configuration" in the *Configuration Reference for Oracle Unified Directory.*
- **Disable Unused Access Logging**. If access logging is not necessary, disabling the server access logger can help improve performance. For more information, see "Log Publisher Configuration" in the *Configuration Reference for Oracle Unified Directory.*
- **Disable Unused Access Control Handlers**. If you do not need access control processing in the server, then you can disable it by setting the `enabled` configuration property to `false` for the Access Control Handler. You can set the property by using `dsconfig`.
- **Reduce Lock Contention**. On systems with large numbers of CPUs (for example, chip multi-threading (CMT) systems with several hardware threads per core), you can reduce lock contention by setting the `org.opends.server.LockManagerConcurrencyLevel` system property to be equal to the number of worker threads you intend to use.

  > **Note:**
  >
  > This property must be set as a JVM system property, because it can be required very early in the server startup process, even before accessing the server configuration.

- **Optimize Operating System Connection Closure**. On a busy LDAP server, if client applications open and close connections at a high rate, the UNIX kernel can run out of connection ports, and client applications will not be able to connect to the *Configuration Reference for Oracle Unified Directory* server. Under these conditions, setting the following property allows the operating system to recycle the connection ports more quickly:

  Linux

  ```
  sysctl -w net.ipv4.tcp_tw_recycle=1
  ```

  Solaris

  ```
  /usr/sbin/ndd -set /dev/tcp tcp_time_wait_interval 30000
  ```

- **Optimize the Virtual Memory Swap Rate**. This ensures that data stays in the filesystem cache longer.

  ```
  sysctl -w vm.swappiness=0
  ```

Part VIII
REST Interfaces

Oracle Unified Directory supports REST API interfaces to Admin, SCIM and Data respectively.

Topics

37

Administering Oracle Unified Directory Using REST API

Oracle Unified Directory allows the users to perform administration and configuration through REST APIs. Admin REST APIs are exposed through HTTP Administration Connector.You can perform basic operations using HTTP methods GET, POST, PATCH or DELETE.

- Configuring Admin REST API
- Invoking the OUD Admin REST API
- Using Admin REST API

37.1 Configuring Admin REST API

You can configure the REST API support for OUD Admin interface during the setup of OUD instance. You need to configure the HTTP Administration Connector port during the setup of OUD instance to expose REST APIs for administering OUD instance.

For more information on HTTP Administration Connector, see HTTP Administration Connector.

Configuring HTTP Administrator Connector Port During OUD Instance Setup

Run `oud-setup` utility from the command line with `httpAdminConnectorPort` parameter to configure the Admin interface while creating the Oracle Unified Directory Server instance.

```
oud-setup --cli
--adminConnectorPort 1444
--httpAdminConnectorPort 1888
--rootUserDN cn=Directory\ Manager
--rootUserPasswordFile password.file
--ldapPort 1389
--ldapsPort 1636
--generateSelfSignedCertificate
--baseDN dc=example,dc=com
--addBaseEntry
--serverTuning jvm-default
--offlineToolsTuning jvm-default
--no-prompt
--noPropertiesFile
```

Configuring HTTP Administration Connector Port for an Existing OUD Instance

Run the `dsconfig` command-line utility with `set-administration-connector-prop` subcommand to update an existing OUD instance to expose HTTP Administration Connector to support Admin REST APIs.

```
dsconfig set-administration-connector-prop  \
--connector-name HTTP \
--set listen-port:1888 \
--set enabled:true \
--hostname localhost\
--port 1444 \
--portProtocol LDAP \
```

```
--trustAll \
--bindDN cn=Directory\ Manager \
--bindPasswordFile password.file \
--no-prompt
```

37.2 Invoking the OUD Admin REST API

You can invoke OUD Admin REST API using the cURL command to send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the specific payload to perform administration tasks.

Following is an example for cURL command to invoke OUD Admin REST API:

```
curl -X POST -k -u '<root User DN>':<Password for root User DN> https://<OUD
Host>:<HTTP Admin Connector Port>/rest/v1/admin -H 'cache-control: no-cache' -
H 'content-type: application/json' -d '<Payload>'
```

37.3 Using Admin REST API

This section includes several sample programs that demonstrate how to perform administrative tasks using the Admin Rest API interface.

- Searching a Network Group
- Adding a Network Group
- Deleting a Network Group
- Comparing a Network Group
- Modifying a Network Group
- Searching a Network Group via GET Method

37.3.1 Searching a Network Group

You can search a particular network group by sending a HTTP request using `POST` method.

To obtain details about a specific network group, send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the following payload:

```
{
"msgType" : "urn:ietf:params:rest:schemas:oracle:oud:1.0:SearchRequest",
"dn" : "cn=network-group,cn=Network Groups,cn=config",
"scope" : "sub",
"filter" : "(objectclass=*)",
"requiredAttributes" : [ "ds-cfg-priority", "ds-cfg-enabled" ],
"base" : "cn=Network Groups,cn=config"
}
```

The following response body is generated when you search for a network group with above mentioned payload:

```
{
    "msgType": "urn:ietf:params:rest:schemas:oracle:oud:1.0:SearchResponse",
    "totalResults": 2,
    "searchResultEntries": [
        {
```

```
            "dn": "cn=Network Groups,cn=config",
            "attributes": {}
        },
        {
            "dn": "cn=network-group,cn=Network Groups,cn=config",
            "attributes": {
                "ds-cfg-priority": "1",
                "ds-cfg-enabled": "true"
            }
        }
    ]
}
```

37.3.2 Adding a Network Group

You can add a particular network group by sending a HTTP request using `POST` method.

To add a specific network group `RestNetworkGroup`, send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the following payload:

```
{
"msgType" : "urn:ietf:params:rest:schemas:oracle:oud:1.0:AddRequest",
"dn" : "cn=RestNetworkGroup,cn=Network Groups,cn=config",
"attributes" : {
"objectclass" : ["top", "ds-cfg-network-group"],
"ds-cfg-priority" : ["0"],
"ds-cfg-enabled" : ["true"],
"cn" : ["RestNetworkGroup"]
}
}
```

The following response body is generated when you add `RestNetworkGroup` using the above mentioned payload:

```
{
    "msgType": "urn:ietf:params:rest:schemas:oracle:oud:1.0:AddResponse",
    "totalResults": 1,
    "searchResultEntries": [
        {
            "dn": "cn=RestNetworkGroup,cn=Network Groups,cn=config",
            "attributes": {
                "ds-cfg-enabled": "true",
                "cn": "RestNetworkGroup",
                "ds-cfg-priority": "0",
                "objectClass": [
                    "top",
                    "ds-cfg-network-group"
                ]
            }
        }
    ]
}
```

37.3.3 Deleting a Network Group

You can delete a particular network group by sending a HTTP request using `POST` method.

To delete a network group, send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the following payload:

```
{
"msgType" : "urn:ietf:params:rest:schemas:oracle:oud:1.0:DeleteRequest",
"dn" : "cn=RestNetworkGroup,cn=Network Groups,cn=config"
}
```

There is no response body generated since this is a delete operation.

37.3.4 Comparing a Network Group

You can compare a particular network group by sending a HTTP request using `POST` method.

To compare a network group, send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the following payload:

```
{
"msgType" : "urn:ietf:params:rest:schemas:oracle:oud:1.0:CompareRequest",
"dn" : "cn=RestNetworkGroup,cn=Network Groups,cn=config",
"assertion" : "ds-cfg-enabled:true"
}
```

The following response body is generated when a compare operation is performed with the above mentioned payload:

```
{
    "msgType": "urn:ietf:params:rest:schemas:oracle:oud:1.0:CompareResponse",
    "compareResult": true
}
```

37.3.5 Modifying a Network Group

You can modify a network group by sending a HTTP request using `POST` method.

To modify a network group, send a request to `https://<OUD HOST>:<HTTP Admin Connector Port>/rest/v1/admin` with the following payload:

```
{
"msgType" : "urn:ietf:params:rest:schemas:oracle:oud:1.0:ModifyRequest",
"operations" :
[
{
"opType" : "replace",
"attribute" : "ds-cfg-enabled",
"values" : ["false"]
}
]
}
```

The following response body is generated when a modify operation is performed with the above mentioned payload:

```
{
    "msgType": "urn:ietf:params:rest:schemas:oracle:oud:1.0:ModifyResponse",
    "totalResults": 1,
    "searchResultEntries": [
        {
            "dn": "cn=RestNetworkGroup,cn=Network Groups,cn=config",
            "attributes": {
                "ds-cfg-enabled": "false",
                "cn": "RestNetworkGroup",
                "ds-cfg-priority": "0",
                "objectClass": [
```

```
                    "top",
                    "ds-cfg-network-group"
                ]
            }
        }
    ]
}
```

37.3.6 Searching a Network Group using GET method

You can search a particular network group by sending an HTTP request to `https://rest/v1/admin/cn=RestNetworkGroup,cn=Network Groups,cn=config` using `GET` method.

No request body for GET.

The following response body is generated when a search operation is performed:

```
{
    "msgType": "urn:ietf:params:rest:schemas:oracle:oud:1.0:SearchResponse",
    "totalResults": 1,
    "searchResultEntries": [
        {
            "dn": "cn=RestNetworkGroup,cn=Network Groups,cn=config",
            "attributes": {
                "ds-cfg-enabled": "true",
                "cn": "RestNetworkGroup",
                "ds-cfg-priority": "0",
                "objectClass": [
                    "top",
                    "ds-cfg-network-group"
                ]
            }
        }
    ]
}
```

38

Managing OUD Directory Data with SCIM REST API

System for Cross-domain Identity Management (SCIM) is a standard protocol for accessing identity information (users, groups, etc) over HTTP(S).

Topics

OUD SCIM interface helps applications in which LDAP is not used to integrate with OUD as their Identity store, or to provision the identity information to OUD.

38.1 Configuring SCIM REST API

You can configure SCIM REST API support for Oracle Unified Directory during the setup.

Oracle Unified Directory exposes SCIM interface through HTTP(S) connection handlers. You can enable these handlers either during an OUD instance setup or through dsconfig for an existing instance.

Configuring Connection Handlers During the OUD Instance Setup

Run the `oud-setup` utility from the command line with `httpPort` and `httpsPort` parameters to configure the SCIM interface while creating the Oracle Unified Directory Server instance.

```
oud-setup --cli \
--adminConnectorPort 1444 \
--httpAdminConnectorPort 1888 \
--rootUserDN cn=Directory\Manager \
--rootUserPasswordFile /home/oracle/pwd.txt \
--ldapPort 1389 \
--ldapsPort 1636 \
--httpPort 1080 \
--httpsPort 1081 \
--generateSelfSignedCertificate \
--baseDN dc=example,dc=com \
--sampleData 200 \
--serverTuning jvm-default \
--offlineToolsTuning jvm-default \
--no-prompt \
--noPropertiesFile
```

Configuring Connection Handlers for an Existing OUD Instance

1. Run the `dsconfig` command-line utility with `create-connection-handler` subcommand as follows to create the connection handlers:

> **Note:**
>
> If you have already created the HTTP/HTTPS connection handler for the OUD instance, then you can update the existing connection handler using the `dsconfig` command-line utility with the `set-connection-handler-prop` subcommand.

Setting Up HTTP Port:

```
dsconfig create-connection-handler \
--handler-name "HTTP Connection Handler" \
--type http \
--set enabled:true \
--set listen-port:1080 \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

Setting Up HTTPS Port:

```
dsconfig create-connection-handler \
--handler-name "HTTPS Connection Handler" \
--type http \
--set enabled:true \
--set listen-port:1081 \
--set use-ssl:true \
--set trust-manager-provider:JKS \
--set key-manager-provider:JKS \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

2. Configure the REST endpoints as follows:
   a. Enable the `REST Server` extension.

```
dsconfig set-extension-prop \
--Extension-name 'REST Server' \
--set enabled:true \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--trustAll \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

   b. Enable the directory endpoint.

```
dsconfig set-directory-end-point-prop \
--set enabled:true \
--hostname localhost \
```

```
--port 1444 \
--portProtocol LDAP \
--trustAll \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

3. Restart the OUD instance.

38.2 Using SCIM REST API

This section provides several sample programs that demonstrate how to make REST API calls through the SCIM interface.

- Creating an Entry
- Modifying an Entry

38.2.1 Creating an Entry

You can create an user entry using SCIM API by sending a HTTP request with `POST` method.

To create an entry through SCIM interface, send a request to URI `/iam/directory/oud/scim/v1/Users` with the following payload.

```
{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:User",
    "urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User",
    "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
  ],
  "name": [
    {
      "formatted": "First name Last name",
      "givenName": " First name ",
      "familyName": " Last name "
    }
  ],
   "password": [
    {
      "value": "password"
    }
  ],
"urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User": {
    "employeenumber": "727",
    "objectClass": [
      {
        "value": "top"
      }
    ],
    "mobile": [
      {
        "value": "+1 503 555 0163"
      }
    ],
    "departmentnumber": [
      {
        "value": "1"
      }
    ]
```

```
  },
  "emails": [
    {
      "value": "First name@example.com"
    }
  ],
  "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User": {
    "department": [
      {
        "value": "1"
      }
    ],
    "employeeNumber": [
      {
        "value": "727"
      }
    ]
  },
  "userName": [
    {
      "value": "First name"
    }
  ]
}
```

The following response body is generated when you create an entry with above mentioned payload:

```
{
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:User",
        "urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User",
        "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
    ],
    "name": [
        {
            "formatted": "First name Last name",
            "givenName": " First name ",
            "familyName": " Last name "
        }
    ],
    "urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User": {
        "objectClass": [
            {
                "value": "top"
            },
            {
                "value": "organizationalPerson"
            },
            {
                "value": "person"
            },
            {
                "value": "inetOrgPerson"
            }
        ],
        "mobile": [
            {
                "value": "+1 503 555 0163"
            }
        ]
    },
```

```
    "meta": {
        "location": "http://localhost:2080/iam/directory/oud/scim/v1/Users/
ad55a34a-763f-358f-93f9-da86f9ecd9e4",
        "resourceType": "User"
    },
    "emails": [
        {
            "value": "First name@example.com"
        }
    ],
    "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User": {
        "department": [
            {
                "value": "1"
            }
        ],
        "employeeNumber": [
            {
                "value": "727"
            }
        ]
    },
    "userName": [
        {
            "value": "First name"
        }
    ],
    "id": "ad55a34a-763f-358f-93f9-da86f9ecd9e4"
}
```

38.2.2 Modifying an Entry

You can modify an user entry using SCIM API by sending a HTTP request with `PATCH` method.

To modify an entry through SCIM interface, send a request to URI `/iam/directory/oud/scim/v1/Users/<Entry UUID>` with the following payload:

> **Note:**
>
> You can search specific entry details by providing entry UUID. This entry UUID is a unique value generated randomly when an entry is created.

```
{
"schemas":
  [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations":
  [
    {
      "op": "replace",
      "path": "urn:ietf:params:scim:schemas:core:2.0:User:password",
      "value": [ "password" ]
    }
  ]
}
```

The following response body is generated when you modify an entry with above mentioned payload


```
{
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:User",
        "urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User",
        "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
    ],
    "name": [
        {
            "formatted": " Replaced First name Last name",
            "givenName": " First name ",
            "familyName": " Last name "
        }
    ],
    "urn:ietf:params:scim:schemas:extension:oracle:2.0:OUD:User": {
        "objectClass": [
            {
                "value": "top"
            },
            {
                "value": "organizationalPerson"
            },
            {
                "value": "person"
            },
            {
                "value": "inetOrgPerson"
            }
        ],
        "mobile": [
            {
                "value": "+1 503 555 0163"
            }
        ]
    },
    "meta": {
        "location": "http://localhost:2080/iam/directory/oud/scim/v1/Users/
ad55a34a-763f-358f-93f9-da86f9ecd9e4",
        "resourceType": "User"
    },
    "emails": [
        {
            "value": "First name@example.com"
        }
    ],
    "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User": {
        "department": [
            {
                "value": "1"
            }
        ],
        "employeeNumber": [
            {
                "value": "727"
            }
        ]
    },
    "userName": [
        {
            "value": "First name"
```

```
                    }
                ]
            }
```

39

Managing Directory Data Using Data Management REST API

You can use the Oracle Unified Directory Data Management REST API to add, modify, remove, and search data in the directory server and manage users and groups.

Topics

39.1 Configuring Data Management REST API

You must configure the OUD environment to use the Oracle Unified Directory Data Management REST API.

Configuring Connection Handlers During the OUD Instance Setup

Run `oud-setup` utility from the command line with `httpPort` and `httpsPort` parameters to configure the Data Management REST API interface while creating the Oracle Unified Directory Server instance.

```
oud-setup --cli \
--adminConnectorPort 1444 \
--httpAdminConnectorPort 1888 \
--rootUserDN cn=Directory\ Manager \
--rootUserPasswordFile password.file \
--ldapPort 1389 \
--ldapsPort 1636 \
--httpPort 1080 \
--httpsPort 1081 \
--generateSelfSignedCertificate \
--baseDN dc=example,dc=com \
--sampleData 200 \
--serverTuning jvm-default \
--offlineToolsTuning jvm-default \
--no-prompt \
--noPropertiesFile
```

Configuring Connection Handlers for an Existing OUD Instance

1. Run the `dsconfig` command-line utility with `create-connection-handler` subcommand as follows to create the connection handlers:

> **Note:**
>
> If you have already created the HTTP/HTTPS connection handler for the OUD instance, then you can update the existing connection handler using the `dsconfig` command-line utility with the `set-connection-handler-prop` subcommand.

Setting Up HTTP Port:

```
dsconfig create-connection-handler \
--handler-name "HTTP Connection Handler" \
--type http \
--set enabled:true \
--set listen-port:1080 \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

Setting HTTPS Port:

```
dsconfig create-connection-handler \
--handler-name "HTTPS Connection Handler" \
--type http \
--set enabled:true \
--set listen-port:1081 \
--set use-ssl:true \
--set trust-manager-provider:JKS \
--set key-manager-provider:JKS \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

2. Configure the REST endpoints as follows:
   a. Enable the `REST Server` extension.

```
dsconfig set-extension-prop \
--Extension-name 'REST Server' \
--set enabled:true \
--hostname localhost \
--port 1444 \
--portProtocol LDAP \
--trustAll \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

   b. Enable the directory endpoint.

```
dsconfig set-directory-end-point-prop \
--set enabled:true \
--hostname localhost \
```

```
--port 1444 \
--portProtocol LDAP \
--trustAll \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--no-prompt
```

3. Restart the OUD instance.

> **Note:**
>
> If the LDAPS connection handler is not configured for the OUD instance, then you must configure `cn=JKS,cn=Key Manager Providers,cn=config` and `cn=JKS,cn=Trust Manager Providers,cn=config` before you set the HTTPS connection handler. See Using JKS Key Manager Provider and Using the JKS Trust Manager Provider

39.2 Using Data Management REST API

The Data Management REST API enables you to to view, add, search, modify or delete directory data in Oracle Unified Directory.

For example scenarios, see *Rest API for Oracle Unified Directory Data Management* .

40

Configuring REST API Support

Learn about the configurations required on OUD Instance for OUD Data Management and SCIM REST API support.

This section contains the following topics:

- Configuring the Server Instance For REST API Support
- Configuring OAM as OAuth Identity Provider in OUD

40.1 Configuring the Server Instance For REST API Support

You can configure the OUD instance for Data Management and SCIM REST API support.

OUD exposes Data Management and SCIM interface through HTTP(S) connection handlers.

> **Note:**
>
> For Admin REST API, see Administering Oracle Unified Directory Using REST API.

Do either of the following to enable the connection handlers:

- Configuring Connection Handlers During the OUD Instance Setup: See Configuring Data Management REST API.
- Configuring Connection Handlers for an Existing OUD Instance: See Configuring Data Management REST API.

40.2 Configuring OAM as OAuth Identity Provider in OUD

OUD supports configuring Oracle Access Management (OAM) as an Open Authorization (OAuth) identity provider.

In addition to the basic authentication technique, OUD supports OAuth 2.0 JWT Access Tokens from OAM for OUD Data Management and SCIM REST API authentication. You can use the JWT token as the Bearer token in calls to OUD Data Management and SCIM REST APIs.

This section discusses the following topics:

- Understanding OAuth Services Authorization
- Configuring OAuth Services

40.2.1 Understanding OAuth Services Authorization

The OAM OAuth 2.0 Service is an open standard OAuth protocol which deals with only delegated authorization.

The following roles are supported by OAuth that helps secure access to protected resources:

- **Resource Server:** The server hosting the protected resources, capable of accepting and responding to resource requests using access tokens. The Resource Server is deployed in a different location from OAM and the Client.
- **Resource Owner:** This is an entity capable of granting access to a protected resource. When the resource owner is a person, it is referred to as an end-user.
- **Client:** It is an application making protected resource requests on behalf of the resource owner and with its authorization.
- **OAuth Services:** Refers to the Authorization Server, Oracle Access Management. The server issuing access tokens to the client after successfully authenticating the resource owner and obtaining authorization.

OAuth 2.0 adds an authorization layer and separates the client's role from that of the resource owner.

In OAuth 2.0, the client seeks access to resources controlled by the resource owner and hosted by the resource server, and is provided credentials that differ from those of the resource owner. To access protected resources, the client acquires an access token - a string identifying a certain scope, duration, and other access attributes - rather than disclosing the credentials of the resource owner. Third-party clients are granted access tokens by an authorization server with the approval of the resource owner. The client utilizes the access token to gain access to the resource server's protected resources. This scenario is referred to as a 3-legged OAuth flow as it involves a resource owner approving a request for resources. See Understanding 3-Legged Authorization.

40.2.2 Configuring OAuth Services

You must first configure the OAuth Services before you can use the authorization protocol.

Perform the following steps to configure the OAuth Services.

1. Setting Up OAuth Services in OAM
2. Setting Up OAM as OAuth Identity Provider in OUD
3. Invoking the REST APIs

40.2.2.1 Setting Up OAuth Services in OAM

This section describes the high-level tasks in setting up OAuth in OAM.

Prerequisite for OAuth Configuration

Before you can configure OUD, you must first configure OAuth Services in OAM and perform the prerequisites mentioned as follows.

1. Create an identity domain using REST API calls, see Creating an Identity Domain.
2. Register a new resource using REST API calls, see Creating a Resource.

> **Note:**
>
> You must configure the resource name and scopes created here in OUD also. For example, a Resource Server named `OUDResourceServer` has been created and the scopes `read` and `write` are set.

3. Create a trusted client using REST API calls, see Creating a Client.

For more information on OAuth REST APIs, See REST API for OAuth in Oracle Access Manager and Managing the Oracle Access Management OAuth Service and OpenIDConnect.

40.2.2.2 Setting Up OAM as OAuth Identity Provider in OUD

You can set up OAM as OAuth Identity Provider in OUD.

Perform the following steps to configure OAM as OAuth Identity Provider in OUD.

1. Run the `dsconfig` command-line utility with `create-identity-provider` subcommand to create OAM as the OAuth Identity Provider in OUD and enable it.

```
dsconfig create-identity-provider  \
--set enabled:true \
--set identity-mapper:Exact\ Match  \
--set oauth-resource-server:OUDResourceServer  \
--set oauth-scope:OUDResourceServer.read  \
--set oauth-token-issuer:http://host4:7777/oauth2  \
--type oauth \
--provider-name OAMProvider \
--hostname localhost \
--port 4444  \
--portProtocol LDAP \
--bindDN "cn=Directory Manager" \
--bindPasswordFile /home/oracle/pwd.txt \
--trustAll \
--no-prompt
```

Table 40-1 OAuth Identity Provider Configuration in OUD

Parameter	Description
`oauth-resource-server`	Refers to the name of the Resource Server during creation in OAM. See Step 2 in Setting Up OAuth Services in OAM.
`oauth-scope`	Refers to the scopes configured for the Resource Server during creation in OAM. This is a multi-valued argument. See Step 2 in Setting Up OAuth Services in OAM. **Note:** The Resource Server name must be prefixed to the scope when the values are provided. For instance, `OUDResourceServer.read`.
`oauth-token-issuer`	Refers to the OAuth Token Issuer URL from OAM. This can be retrieved using the OpenIDConnect Discovery Endpoint from OAM. For more information on the Discovery Endpoint, see OpenIDConnect Authentication Flows in Oracle Access Manager.
`identity-mapper`	An Identity Mapper needs to be configured to map the user ("sub" claim) in the JWT Token to a valid user in OUD. To create an identity mapper, see Overview of Identity Mappers.
`oauth-provider`	Refers to the name of the OAuth 2.0 compliant Identity Provider. Currently OUD only supports OAM. Therefore, the default value is `OAM`.

Table 40-1 (Cont.) OAuth Identity Provider Configuration in OUD

Parameter	Description
`oauth-token-x5t-algorithm`	Refers to the algorithm that is used to generate the thumbprint of an Identity Provider's X.509 certificate. **Note:**The access token header should include the thumbprint generated using the same algorithm. The default value is `SHA-1`, which assumes that the access token header contains an `x5t` property with the value set to the `SHA-1` fingerprint of the certificate. However, if the access token issued by the Identity provider lacks the `x5t` property and only contains the `x5t#256` property, the value here should be `SHA-256`.
`token-introspection-endpoint`	Refers to the URL for the OAuth token introspection endpoint. This is an optional parameter. When configured, the OAuth Server is also queried to ensure that the JWT Token is genuine. To obtain the OAM OAuth token introspection endpoint, use the OpenIDConnect Discovery Endpoint. This must be a POST operation. For instance, `http://host4:7777/oauth2/rest/token/introspect`.
`custom-claim-to-id-mapper`	Refers to the name of the custom claim in the Access Token that will be utilized by the identity-mapper to map to a valid user record in OUD. This is an optional parameter, which if not configured, `sub` in the token claim is mapped automatically.

2. Set the `http-authentication-scheme` in REST Server Extension to include the bearer in addition to basic scheme.

```
dsconfig set-extension-prop \
--extension-name REST\ Server  \
--add http-authentication-scheme:bearer \
--hostname localhost \
--port 4444  \
--portProtocol LDAP \
--bindDN "cn=Directory Manager"  \
--bindPasswordFile /scratch/nenekris/OUD/pwd.txt \
--trustAll \
--no-prompt
```

> **Note:**
>
> The REST Server Extension is enabled by default; if it is not, see Configuring the Server Instance For REST API Support.

3. Import the public key certificates (JWT token signing keys) of OAM into OUD's JKS truststore so that OUD can perform signature validation of the JWT Tokens when presented in a REST API call.

> **Note:**
>
> The OAM's trust certificate can be obtained using OpenIDConnect Discovery Endpoint using the jwks_uri: `http://<ManagedServerHost>:<ManagedServerPort>/oauth2/rest/security`
> For more information on the Discovery Endpoint, see OpenIDConnect Authentication Flows in Oracle Access Manager.

40.2.2.3 Invoking the REST APIs

You can invoke the Data Management or SCIM REST APIs for OAuth.

This following sample REST requests show how to invoke the OUD Data Management REST API using the cURL command to send a request to `https://<OUD HOST>:<HTTP Port>/rest/v1/directory` with a specific payload.

With basic authentication:

```
curl -X GET -u '<userdn>':<password> http://<OUD Host>:<HTTP Port>/rest/v1/
directory/dc=example,dc=com?scope=sub&filter=(objectclass=*) -H 'cache-
control: no-cache' -H 'content-type: application/json'
```

With Bearer token in the header:

```
curl -X GET -H "Authorization: Bearer {token}" http://<OUD Host>:<HTTP Port>/
rest/v1/directory/dc=example,dc=com?scope=sub&filter=(objectclass=*) -H
'cache-control: no-cache' -H 'content-type: application/json'
```

A

Appendixes and Glossary

You may need to consult this supplemental information like the command line interface reference, the LDAP reference and the standards and specifications supported by Oracle Unified Directory.

This part contains the following appendixes and a glossary:

- Oracle Unified Directory Command-Line Interface Reference
- LDAP Controls and Operations Reference
- Standards and Specifications Supported by Oracle Unified Directory

A.1 Oracle Unified Directory Command-Line Interface Reference

Follow these topics for a description of the command-line utilities used by Oracle Unified Directory to create, configure, and manage directory server, proxy server and replication gateway instances.

- General Command-Line Usage Information
- Server Administration Commands
- Data Administration Commands
- LDAP Client Commands

This appendix describes all of the commands that are provided with Oracle Unified Directory. Some commands are specific to a directory server instance and cannot be used to configure a proxy server. Similarly, some commands are specific to the proxy and cannot be used to configure a directory server.

A.1.1 General Command-Line Usage Information

Review these topics for general information on server commands usage.

- Summary of Server Commands and Their Use
- Using a Properties File With Server Commands
- Using a Password File With Server Commands
- Managing CLI Log Configuration for Server Commands

A.1.1.1 Summary of Server Commands and Their Use

The tables in this section provide a summary of the server commands and how they can be used. The tables use the following legend:

Remote
The command can be launched on a remote server

Offline
The command can be launched when the server is stopped

Online
The command connects to a running server instance

Administration Port Only
The command *must* use the administration connector to access the server (on port 4444 by default)

> **Note:**
>
> Not all the commands listed in the following tables are supported for a proxy server instance.

The following table lists the server administration commands:

Table A-1 Server Administration Commands

Command	Remote	Offline	Online	Administration Connector
create-rc-script				
dsconfig	X		X	X
dsjavaproperties		X		
dsreplication	X		X	X
gicadm	X		X	X
oudExtractMovePlan		X	X	
oudCopyConfig		X	X	
oudPasteConfig		X		
start-ds		X		
status	X	X	X	X
stop-ds	X		X	X
uninstall		X	X	X
upgrade		X		
windows-service		X		

The following table lists the data administration commands:

Table A-2 Data Administration Commands

Command	Remote	Offline	Online	Administration Connector
backup	X *	X	X	X
base64		X		
dbtest		X		
encode-password		X		

Table A-2 (Cont.) Data Administration Commands

Command	Remote	Offline	Online	Administration Connector
export-ldif	X *	X	X	X
import-ldif	X *	X	X	X
ldapcompare	X		X	
ldapdelete	X		X	
ldapmodify	X		X	
ldappasswordmodify	X		X	
ldapsearch	X		X	
ldif-diff		X		
ldifmodify		X		
ldifsearch		X		
list-backends		X		
make-ldif		X		
manage-account	X		X	X
manage-tasks	X		X	X
purge-backup	X *	X	X	X
rebuild-index		X		
restore	X *	X	X	X
split-ldif		X	X	
verify-index		X		

* The command can be launched remotely but the data files must be on the host on which the server is running.

A.1.1.2 Using a Properties File With Server Commands

Certain command-line utilities can use a common properties file to provide default values for options such as the following:

- The host name and port number of the server
- Whether to use SSL or StartTLS to communicate with the server
- The bind DN to use when connecting to the server

A.1.1.2.1 Utilities That Can Use Properties Files

The following utilities can use a properties file:

- `backup`
- `dsconfig`
- `dsreplication`
- `export-ldif`

- gicadm
- import-ldif
- split-ldif
- ldapcompare
- ldapdelete
- ldapmodify
- ldappasswordmodify
- ldapsearch
- manage-tasks
- oud-setup
- oud-proxy-setup
- oud-replication-gateway-setup
- restore
- status
- stop-ds
- uninstall

The following mutually exclusive options are used with the command-line utilities to indicate whether a properties files is used:

`--propertiesFilePath` ***path***
Specify the path to the file that contains default values for command-line options.

`--noPropertiesFile`
Indicates that the properties file is not used to obtain default values for command-line options.

A.1.1.2.2 How Properties Files are Located

Utilities that use the common properties file have the following default behavior:

- If the `--noPropertiesFile` option is specified, the command-line interface does not try to locate a properties file. Only options specified on the command line are evaluated.
- If the `--propertiesFilePath` option is specified, property values are read from this file.
- If neither `--propertiesFilePath` nor `--noPropertiesFile` is specified, the command-line interface attempts to find a properties file in the following locations:
  - *USERDIRECTORY*`/.opends/tools.properties`
  - *INSTANCE_DIR*`/OUD/config/tools.properties`
- If no properties file is found in either of these locations, the default behavior is applied (only arguments specified on the command line are evaluated).

A.1.1.2.3 Order of Precedence of Options and Properties

If an option is provided on the command line, this option and its corresponding value are used by the command-line interface. In other words, options specified on the command line take precedence over the properties defined in the properties file.

The properties file has the standard JAVA properties file format (*property-name*=*value*). As such, the file supports variations on property names to enable them to be overridden according to the command that uses them. For example, the properties file might contain the following:

```
hostname=localhost
port=4444
bindDN=cn=Directory Manager
bindPasswordFile=/path/pwd-file
baseDN=dc=example,dc=com
searchScope=sub
sortOrder=givenName
virtualListView=0:2:1:0
```

If a command-line interface uses the `port` property, the command first tries to locate a *toolname*`.port` definition. If this is not defined, the command tries to locate a `port` definition. For example, the properties file might have several port options defined for different utilities:

```
port=4444
ldapsearch.port=1389
ldapcompare.port=1389
ldapmodify.port=1389
ldapdelete.port=1389
```

> **Note:**
>
> Do **not** use quotation marks around the values in the properties file (for example, `port="4444"`).

A.1.1.3 Using a Password File With Server Commands

Certain command-line utilities require a password file that contains only the password for the user account or entry (`bindDN`) using which bind is performed. You use the bind password in the specified file instead of passing it in clear text form on the command line.

Perform the following steps to create a text file, for instance `pwd-file`, to save the `bindDN` password.

1. Navigate to the location where you want to create a text file to store the password.
2. Create the text file that will only contain the password as follows:

   ```
   vi pwd-file
   ```

3. Enter the password.
4. Save and close the file.

You must use this password file, `pwd-file`, while specifying the `-j` or the `--bindPasswordFile` parameter for command-line utilities, such as `dsconfig`, `dsreplication`, `oud-setup`, and so on.

You must always specify the absolute or relative path to the password file if it does not exists in your current directory.

A.1.1.4 Managing CLI Log Configuration for Server Commands

Some server administration commands, such as `dsreplication` and `status`, generate client-side log files called *oud-replication-IDnumber* and *oud-status-IDnumber*, where *`IDnumber`* is a decimal number.

You can find the log files at the following location:

- UNIX (Solaris): `/var/tmp/`
- Linux: `/tmp/`
- Windows: `%TEMP%`
  By default, the log file folder is `C:\Documents and Settings\User\Local Settings\Temp`.

You can use the following JVM arguments to configure the generation and location of the log files:

- `-Dcli.log.level`: Sets the level of logging for the `dsreplication` and `status` CLI tools. Valid Values are: `OFF`, `SEVERE`, `WARNING`, `INFO`, `CONFIG`, `FINE`, `FINER`, `FINEST`, `ALL`. To disable logging, use the level value `OFF`.
- `-Dcli.log.location`: Replaces the default log file location with a custom location where the log file must be written.

Follow the procedure in the example section of A.1.2.5 dsjavaproperties to change the `status` and `dsreplication` CLI settings.

Example 1

To disable the `status` CLI log, modify java.properties as follows:

```
status.java-args=-client -Dcli.log.level=OFF
```

Example 2

To redirect the `dsreplication` CLI log to a custom location, modify java.properties as follows:

```
dsreplication.java-args=-client -Dcli.log.location=/scratch/OUD_CLI_LOGS
```

A.1.2 Server Administration Commands

You can review the different options and examples of each server administration command.

- create-rc-script
- dps2oud
- ds2oud
- dsconfig
- dsjavaproperties
- dsreplication
- dstune
- gicadm
- manage-tasks
- oudCopyConfig
- oudExtractMovePlan
- oudPasteConfig
- oud-replication-gateway-setup

- oud-setup
- oud-proxy-setup
- start-ds
- status
- stop-ds
- uninstall
- windows-service

A.1.2.1 create-rc-script

The `create-rc-script` command generates a shell script to start, stop, and restart the directory server.

Synopsis

`create-rc-script` [*options*]

Description

The `create-rc-script` command can be used to generate a shell script to start, stop, and restart the directory server. You can update the resulting script to suit the needs of your directory service. This command is available for UNIX or Linux systems only.

The `create-rc-script` command uses the `OPENDS_JAVA_*` and `JAVA_*` variables.

Options

The `create-rc-script` command accepts an option in either its short form (for example, `-f` *filename*) or its long form equivalent (for example, `--outputFile` *filename*).

`-f, --outputFile` ***filename***
Specify the path to the output file.

`-j, --javaHome` ***javaHomePath***
Specify the path to the Java installation that should be used to run the server.

`-J, --javaArgs` ***javaArgs***
Specify the set of arguments that should be passed to the JVM when running the server.

`-u, --userName` ***userName***
Specify the name of the user account under which the server should run. The user account must have the appropriate permissions to run the script.

General Options

`--version`
Display the version information for the directory server.

`-?, -H, --help`
Display command-line usage information for the `create-rc-script` command.

Examples

The examples in this section explain how to use the `create-rc-script` command.

Creating the Script

The following command generates the script to start, stop, and restart the directory server. It creates the file called `myscript`, specified by the `-f` option:

```
$ create-rc-script -f myscript
```

Starting the Directory Server by Using the New Script

The following command uses the newly created script (see previous example) to start the directory server.

```
$ myscript start
```

Stopping the Directory Server by Using the New Script

The following command uses the newly created script (see first example) to stop the directory server.

```
$ myscript stop
```

Restarting the Directory Server by Using the New Script

The following command uses the newly created script (see first example) to restart the directory server.

```
$ myscript restart
```

Specifying JAVA_HOME and JAVA_ARGS in the Script

The following command uses the `-u` (`--userName`), `-j` (`--javaHome`) and `-J` (`--javaArgs`) options.

```
$ create-rc-script -f myscript -u sysAdmin -j /usr/java -J "-Xms128m -Xmx128m"
```

Code Generated by the `create-rc-script` Command

The `create-rc-script` command from the example above generates the following code:

```
# /bin/sh
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License, Version 1.0 only
# (the "License").  You may not use this file except in compliance
# with the License.
#
# You can obtain a copy of the license at
# https://OpenDS.dev.java.net/OpenDS.LICENSE.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at
# trunk/opends/resource/legal-notices/OpenDS.LICENSE.  If applicable,
# add the following below this CDDL HEADER, with the fields enclosed
# by brackets "[]" replaced with your own identifying information:
#      Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END

# Set the path to the OpenDS instance to manage
```

```
INSTANCE_ROOT="/usr/local/opends/standalone/ds-server-1"
export INSTANCE_ROOT

# Specify the path to the Java installation to use
OPENDS_JAVA_HOME="/usr/java"
export OPENDS_JAVA_HOME

# Specify arguments that should be provided to the JVM
JAVA_ARGS="-Xms128m -Xmx128m"
export JAVA_ARGS

# Determine what action should be performed on the server
case "${1}" in
start)
/bin/su sysAdmin "${INSTANCE_ROOT}/bin/start-ds" --quiet
exit ${?}
;;
stop)
/bin/su sysAdmin "${INSTANCE_ROOT}/bin/stop-ds" --quiet
exit ${?}
;;
restart)
/bin/su sysAdmin "${INSTANCE_ROOT}/bin/stop-ds" --restart --quiet
exit ${?}
;;
*)

echo "Usage:  $0 { start | stop | restart }"
exit 1
;;
esac
```

Exit Codes

An exit code of 0 indicates success. A nonzero exit code indicates that an error occurred.

Location

The `create-rc-script` command is located at this path:

UNIX and Linux: *INSTANCE_DIR*`/OUD/bin`

Related Commands

- start-ds
- stop-ds

A.1.2.2 dps2oud

The `dps2oud` command allows you to migrate a Directory Proxy Server (DPS) configuration to an Oracle Unified Directory configuration.

Synopsis

`dps2oud` [*options*]

Description

The `dps2oud` command allows you to migrate a DPS configuration to an Oracle Unified Directory configuration. The `dps2oud` command takes a DPS configuration as the input and

generates a batch file that comprises `dsconfig` commands, which are used to create an equivalent Oracle Unified Directory configuration. The `dps2oud` command reads the DPS configuration either through a file or through the LDAP protocol on a running DPS instance.

Options

The `dps2oud` command accepts the following options.

`-o, --outputFile` ***file***
The output file for `dsconfig` commands.

`-f, --dpsConfigFile` ***file***
Specifies the name of the DPS config file to use.

`-c, --createDisabledObjects`
Creates DPS-disabled objects.

`-P, --printDsConfigCmds`
Prints `dsconfig` commands.

LDAP Connection Options

`-h, --hostname` ***host***
DPS server hostname or IP address.

`-j, --bindPasswordFile` ***filename***
The full path to the file containing the bind password.

`-p, --port` ***port***
DPS server port number.

`-D, --BindDN` ***bindDN***
DN to use to bind to the DPS server.

General Options

`-?, -H, --help`
Displays command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

`-V, --version`
Displays the version information for the directory server.

Examples

The following examples show how to use the `dps2oud` command.

Viewing the Global Help Subcommands

The following command displays the available global Help subcommands:

```
$ dps2oud --help
```

Migrating a Directory Proxy Server Configuration to an Oracle Unified Directory Configuration

You can migrate a DPS configuration to an Oracle Unified Directory configuration using one of the following methods:

Method 1: Reading a DPS configuration from an LDIF file

The following command displays how to read a DPS configuration from an LDIF file:

```
$ dps2oud -f dse.ldif -o oud_conf_cmds
```

The following command provides the path to a batch file containing a set of `dsconfig` commands to be executed:

```
$ dsconfig -F oud_conf_cmds
```

Method 2: Reading a DPS configuration from a running DPS instance

The following command displays how to read a DPS configuration from a DPS instance:

```
$ dps2oud -h dpsHost -p 389 -D "cn=Proxy Manager"  -j /path/pwd-file -o oud_conf_cmds
```

The following command provides the path to a batch file containing a set of `dsconfig` commands to be executed:

```
$ dsconfig -F oud_conf_cmds
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dps2oud`
- Windows: *INSTANCE_DIR*`\OUD\bat\dps2oud.bat`

Related Commands

- dsconfig

A.1.2.3 ds2oud

The `ds2oud` command manages the migration from an Oracle Directory Server Enterprise Edition directory server instance to Oracle Unified Directory.

Synopsis

`ds2oud` [*options*]

Description

The `ds2oud` command enables you to manage the migration from an Oracle Directory Server Enterprise Edition directory server instance to Oracle Unified Directory. The `ds2oud` command first allows you to diagnose the targeted Oracle Directory Server Enterprise Edition directory server, and then performs the migration task. It is based on the premise that the existing Oracle Unified Directory instance is modified to be compatible with the Oracle Directory Server Enterprise Edition directory server to be migrated. The `ds2oud` command runs in interactive mode, if you do not specify options. Interactive mode works much like a wizard, walking you through every aspect of the migration.

You can also run the `ds2oud` command in batch mode. In batch mode, a batch file that comprises `dsconfig` commands is generated. These commands are used to create an equivalent Oracle Unified Directory configuration. So, you can run `ds2oud` once, and create a single batch file that can be used to configure any number of Oracle Unified Directory instances.

You must ensure while running the `ds2oud` command that the Oracle Unified Directory instance (to which the Oracle Directory Server Enterprise Edition instance is being migrated) is configured without any suffixes.

Options

The `ds2oud` command accepts the following options.

`-d, --diagnose`
Diagnoses the targeted Oracle Directory Server Enterprise Edition directory server.

`-f, --ldifDBFile` ***file***
Diagnoses the Oracle Directory Server Enterprise Edition directory server LDIF database file.

`-u, --userSchemaFile` ***file***
Specifies the user schema to be taken into consideration. It applies to `-f` subcommand.

`-a, --migrateAll`
Propagates schema and configuration elements from Oracle Directory Server Enterprise Edition directory server to Oracle Unified Directory server.

`-s, --migrateUserSchema`
Propagates the User schema from Oracle Directory Server Enterprise Edition directory server to Oracle Unified Directory server.
You must migrate the schema *before* you migrate the configuration, otherwise the migration can produce unpredictable results.

`-c, --migrateConfiguration`
Propagates configuration elements from Oracle Directory Server Enterprise Edition directory server to Oracle Unified Directory server.
You must migrate the schema *before* you migrate the configuration, otherwise the migration can produce unpredictable results.

`-A, --adaptDseeData {file}`
Adapts an Oracle Directory Server Enterprise Edition 6.3 LDIF data file to ease import on Oracle Unified Directory, producing a result file, `{`*file*`}_result.ldif`

`-w, --uniqueWorkflowElement`
Use a unique workflow element for all the naming contexts to migrate. This applies to `-c` subcommand.

Oracle Directory Server Enterprise Edition LDAP Connection Options

`-D, --odseeBindDN` ***bindDN***
DN to use to bind to the Oracle Directory Server Enterprise Edition server.

`-j, --odseeBindPasswordFile` ***filename***
Oracle Directory Server Enterprise Edition bind password file.

`-h, --odseeHostname` ***host***
Oracle Directory Server Enterprise Edition server hostname. The default value is localhost.

`-p, --odseePort` ***port***
Oracle Directory Server Enterprise Edition server port number. The default value is 389.

`-Z, --odseeUseSSL`
Establishes an Oracle Directory Server Enterprise Edition SSL-encrypted connection.

`-P, --odseeTrustStorePath` ***trustStorePath***
Use the Oracle Directory Server Enterprise Edition trust store certificate in the specified path. This option is not needed if `-X` is used, although a trust store should be used when working in a production environment.

`-U, --odseeTrustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the Oracle Directory Server Enterprise Edition trust store. This option is only required if `--odseeTrustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --odseeTrustAll`
Trust all certificate that the Oracle Directory Server Enterprise Edition server presents. This option can be used for testing purposes, but for security reasons, a trust store should be used to determine whether the Oracle Directory Server Enterprise Edition should accept the server certificate.

Oracle Unified Directory LDAP Connection Options

`--oudBindDN` ***bindDN***
DN to use to bind to the Oracle Unified Directory server.

`--oudBindPasswordFile` ***filename***
Oracle Unified Directory bind password file.

`--oudHostname` ***host***
Oracle Unified Directory server hostname. The default value is localhost.

`--oudPort` ***port***
Oracle Unified Directory server port number. The default value is 389.

`--oudAdminPort` ***port***
Oracle Unified Directory server administration port. The default value is 444.

`--oudUseSSL`
Establishes an Oracle Unified Directory SSL-encrypted connection.

`--oudTrustStorePath` ***trustStorePath***
Use the Oracle Unified Directory trust store certificate in the specified path.

`--oudTrustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the Oracle Unified Directory trust store. This option is only required if `--oudTrustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`--oudTrustAll`
Trust all certificate that the Oracle Unified Directory server presents. This option can be used for testing purposes, but for security reasons, a trust store should be used to determine whether the Oracle Unified Directory should accept the server certificate.

Command Input/Output Options

`-n, --no-prompt`
Use the non-interactive mode. If data in the command is missing, the user is not prompted and the tool fails.

`-o, --outputFile` *filename*
Redirects the output into the specified output file.

`-F, --batchFilePath` *filename*
This option specifies the name of the output file that contains a set of `dsconfig` commands to execute to migrate the configuration.
When you run `ds2oud` with this option, a batch file is generated that includes all of the `dsconfig` commands required to create the equivalent Oracle Unified Directory configuration. So, you can run `ds2oud` once, and create a single batch file that can be used to configure any number of Oracle Unified Directory instances.

`--displayCommand`
Display the equivalent non-interactive dsconfig commands (for the migration of Oracle Directory Server Enterprise Edition configuration parameters).

General Options

`-?, -H, --help`
Displays command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

`-V, --version`
Displays the version information for the directory server.

Examples

The following examples show how to use the `ds2oud` command.

Viewing the Global Help Subcommands

The following command displays the available global Help subcommands:

```
$ ds2oud --help
```

Running `ds2oud` in Interactive Mode From the Command Line

The `ds2oud` command can be run in interactive mode, where you are prompted for migration options. To run `ds2oud` in interactive mode, type the following command:

```
$ ds2oud
What do you want to do ?

1)  Diagnose an ODSEE directory server instance
2)  Diagnose an ODSEE LDIF data file
3)  Migrate the user schema and global configuration parameters
4)  Migrate the user schema only
5)  Migrate global configuration parameters only
6)  Adapt DSEE 6.3 LDIF data file to ease import on OUD

c)  cancel
```

For each preceding action, you must first provide the connection options for the Oracle Directory Server Enterprise Edition server (for diagnosis) or both the Oracle Directory Server Enterprise Edition and Oracle Unified Directory servers (for migration).

Running `ds2oud` for Diagnosing Data

The following command is run to diagnose the data present in the Oracle Directory Server Enterprise Edition directory server:

```
$ ds2oud -f odseeDataFile.ldif -u 99-user.ldif

*******************************************************************************
* Diagnose ODSEE LDIF data file :  odseeDataFile.ldif
*******************************************************************************
The data were validated successfully regarding the OUD schema
```

Migrating an Existing Oracle Directory Server Enterprise Edition Configuration to an Oracle Unified Directory Configuration

Use the following commands to migrate an existing Oracle Directory Server Enterprise Edition Configuration to a new Oracle Unified Directory Configuration

The following command migrates an existing Oracle Directory Server Enterprise Edition configuration and schema:

```
$ ds2oud --migrateAll -D "cn=directory manager"
-j /tmp/pwd -h hostname -p ldapPort
--oudBindDN "cn=directory manager" --oudBindPasswordFile /tmp/pwd
--oudHostname hostname2 --oudPort ldapPort2 --oudAdminPort adminPort  -n
```

The following command provides the path to a batch file containing a set of `dsconfig` commands to be executed to create a new Oracle Unified Directory configuration:

```
$ ds2oud --migrateConfiguration --batchFilePath batchFile
-D "cn=directory manager" -j /tmp/pwd -h hostname
-p ldapPort --oudBindDN "cn=directory manager"
--oudBindPasswordFile /tmp/pwd  --oudHostname hostname2
--oudPort ldapPort2 --oudAdminPort adminPort -n
```

Exit Codes

0
Successful.

1
Unable to initialize arguments.

2
Cannot parse arguments because the provided arguments are not valid or there was an error checking the user data.

3
At least one step into the migration process has failed.

4
The user canceled the operation in interactive mode.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ds2oud`
- Windows: *INSTANCE_DIR*`\OUD\bat\ds2oud.bat`

Related Commands

- dsconfig

A.1.2.4 dsconfig

The `dsconfig` command allows you to define a base configuration for the Directory Server.

Synopsis

`dsconfig` [*subcommands*] [*Options*]

Description

The `dsconfig` command enables you to create, manage, and remove the base configuration for a server instance. The server configuration is organized as a set of components that `dsconfig` can access by using one or more subcommands. All components have zero or more configurable properties. These properties can be queried and modified to change the behavior of the component.

The `dsconfig` command accesses the server over SSL through the administration connector (described in Managing Administration Traffic to the Server).

Unless you specify all configuration parameters and the `-n` (`--no-prompt`) option, `dsconfig` runs in interactive mode. Interactive mode works much like a wizard, walking you through every aspect of the server configuration. For more information, see Using `dsconfig` in Interactive Mode.

- Help Subcommands
- General Subcommands
- Distribution Subcommands
- General Configuration Subcommands
- Load Balancing Subcommands
- Local Data Source Subcommands
- Integration Subcommands
- Remote Data Source Subcommands
- Replication Subcommands
- Schema Subcommands
- Security Subcommands
- Virtualization Subcommands

Help Subcommands

The `dsconfig` command provides help functions that list the component subcommands needed to manage your base configuration.

`--help-distribution`
Display subcommands relating to distribution.

`--help-general-configuration`
Display subcommands relating to general configuration.

`--help-integration`
Display subcommands relating to integration.

`--help-load-balancing`
Display subcommands relating to load balancing.

`--help-local-datasource`
Display subcommands relating to local data source.

`--help-remote-datasource`
Display subcommands relating to remote data source.

`--help-replication`
Display subcommands relating to replication.

`--help-schema`
Display subcommands relating to schema.

`--help-security`
Display subcommands relating to authentication and authorization.

`--help-virtualization`
Display subcommands relating to virtualization.

`--help-all`
Display all subcommands.

General Subcommands

The following subcommand lists the objects and properties of the server instance.

list-properties
Displays the managed objects and properties. Option types are as follows:
`r` — Property values are readable.
`w` — Property values are writable.
`m` — The property is mandatory.
`s` — The property is single-valued.
`a` — Administrative action is required for changes to take effect.
Suboptions are as follows:
`-t, --type` *type*. Component type.
`-c, --category` *category*. Category of the component. The value for `type` must be one of the component types associated with the *category* that is specified using the `--category` suboption.
`--inherited`. Modifies the display output to show the inherited properties of components.
`--advanced`. Modifies the display output to show the advanced properties of components.
`--property` *property*. The name of a property to be displayed.

Distribution Subcommands

The following subcommands allow you to define the base configuration for the directory server.

`create-distribution-algorithm`
Creates distribution algorithms. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Distribution Algorithm that should be created. The value for *type* can be one of `capacity, dnpattern, generic, lexico,` or `numeric`.

`create-distribution-partition`
Creates distribution partitions. Suboptions are as follows:

`--element-name` *name*. The name of the distribution workflow element.
`--partition-name` *name*. The name of the new distribution partition.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Distribution Partition that should be created. The value for *type* can be one of `capacity`, `dnpattern`, `generic`, `lexico`, or `numeric`.

`create-workflow-element --type distribution`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the new Workflow Element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type.* The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of`

`create-global-index`
Creates global indexes. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--index-name` *name.* The name of the new Global Index.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-extension --type global-index-catalog`
Creates Extensions. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type.* The type of Extension that should be created. The value for type can be one of `global-index-catalog, global-index-catalogs-shared-cache, ldap-server.`

`create-global-index-catalog-replication-domain`
Creates global index catalog replication domains. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-extension --type global-index-catalogs-shared-cache`
Creates Extensions. Suboptions are as follows:
`--extension-name` *name*. The name of the new Extension.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`-t,--type` *type.* The type of Extension that should be created. The value for *type* can be one of `global-index-catalog, global-index-catalogs-shared-cache, ldap-server.`

`create-workflow-element --type global-index-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the new Workflow Element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type.* The type of Workflow Element that should be created. The value for *type* can be one of `ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, chema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of.`

`create-workflow-element --type global-index-replication-changes-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the new Workflow Element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type.* The type of Workflow Element that should be created. The value for *type* can be one of `ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, chema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of.`

`delete-distribution-algorithm`
Deletes distribution algorithms. Suboptions are as follows:
`--element-name` *name*. The name of the Distribution Workflow Element.
`-f,--force`. Ignore nonexistent distribution algorithms.

`delete-distribution-partition`
Deletes distribution partitions. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--partition-name` *name*. The name of the distribution partition.
`-f,--force`. Ignore nonexistent distribution partitions.

`delete-extension`
Deletes Extensions. Suboptions are as follows:
`--extension-name` *name*. The name of the Extension.
`-f,--force`. Ignore nonexistent extensions.

`delete-global-index`
Deletes global indexes. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--index-name` *name.* The name of the Global Index.
`-f,--force`. Ignore nonexistent global indexes.

delete-global-index-catalog-replication-domain
This command is supported only for the proxy. To manage the global index see gicadm
Deletes global index catalog replication domains. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`-f,--force`. Ignore nonexistent global index catalog replication domains.

delete-workflow-element
Deletes Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the Workflow Element.
`-f,--force`. Ignore nonexistent workflow element.

get-data-encryption-prop
Shows data encryption properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-distribution-algorithm-prop
Shows distribution algorithm properties. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-distribution-partition-prop
Shows distribution partition properties. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--partition-name` *name*. The name of the distribution partition.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-global-index-catalog-replication-domain-prop
This command is supported only for the proxy. To manage the global index see gicadm
Shows global index catalog replication domain properties. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-global-index-prop
This command is supported only for the proxy. To manage the global index see gicadm

Shows Global index properties. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--index-name` *name.* The name of the Global Index.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-distribution-algorithm`
This command is supported for only proxy.
Lists existing distribution algorithm. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-distribution-partitions`
This command is supported only for the proxy.
Lists existing distribution partitions. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-extensions`
Lists existing Extensions. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-global-index-catalog-replication-domain`
This command is supported only for the proxy. To manage the global index see gicadm
Lists existing global index catalog replication domain. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-global-indexes`
Lists existing global indexes. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing Workflow Elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-data-encryption-prop`
Modifies Data Encryption properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-distribution-algorithm-prop`
This command is supported only for the proxy.
Modifies distribution algorithm properties. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-distribution-partition-prop`
This command is supported only for the proxy.
Modifies distribution partition properties. Suboptions are as follows:
`--element-name` *name*. The name of the distribution workflow element.
`--partition-name` *name*. The name of the distribution partition.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-extension-prop`
Modifies Extension properties. Suboptions are as follows:
--extension-name *name*. The name of the Extension.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-global-index-catalog-replication-domain-prop`
This command is supported only for the proxy.
Modifies global index catalog replication domain properties. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-global-index-prop`
This command is supported only for the proxy.
Modifies global index properties. Suboptions are as follows:
`--extension-name` *name*. The name of the Global Index Catalog Extension.
`--index-name` *name.* The name of the Global Index.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
Modifies Workflow Element properties. Suboptions are as follows:
`--element-name` *name.* The name of the Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

General Configuration Subcommands

The following subcommands configure the core server.

create-alert-handler
Creates alert handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the new alert handler.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Alert Handler that should be created. The value for *type* can be one of `custom`, `jmx`, or `smtp`.

create-certificate-mapper
Creates certificate mappers. Suboptions are as follows:
`--mapper-name` *name*. The name of the new certificate mapper.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Certificate Mapper that should be created. The value for *type* can be one of `custom`, `fingerprint`, `subject-attribute-to-user-attribute`, `subject-dn-to-user-attribute`, or `subject-equals-dn`.

create-connection-handler
Creates connection handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the new connection handler.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Connection Handler that should be created. The value for *type* can be one of `custom`, `jmx`, `ldap`, `snmp`, or `ldif`.

create-debug-target
Creates debug targets. Suboptions are as follows:
`--publisher-name` *name*. The name of the debug log publisher.
`--target-name` *java-name*. The name of the new debug target, which will also be used as the value for the `debug-scope` property. The fully-qualified Oracle Unified Directory Java package, class, or method affected by the settings in this target definition. Use the hash symbol (`#`) to separate the class name and the method name (for example, `org.opends.server.core.DirectoryServer#startUp`).
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

create-extended-operation-handler
This command is not supported for the proxy.
Creates extended operation handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the new extended operation handler.
`--set property:value`. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Extended Operation handler that should be created. The value for *type* can be one of `cancel`, `custom`, `get-connection-id`, `get-symmetric-key`, `password-modify`, `password-policy-state`, `start-tls`, or `who-am-i`.

create-identity-mapper
Creates identity mappers. Suboptions are as follows:
--mapper-name *name*. The name of the new identity mapper.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Identity Mapper that should be created. The value for *type* can be one of `custom`,`exact-match`, or `match-and-replace`.

`create-log-publisher`
Creates log publishers. Suboptions are as follows:
`--publisher-name` *name*. The name of the new log publisher.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Log Publisher that should be created. The value for *type* can be one of `custom-access, custom-debug, custom-error, file-based-access`, `file-based-debug`, or `file-based-error`.

`create-log-retention-policy`
Creates Log Retention Policies. Suboptions are as follows:
`--policy-name` *name*. The name of the new log retention policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Log Retention Policy that should be created. The value for *type* can be one of `custom`,`file-count`, `free-disk-space`, or `size-limit`.

`create-log-rotation-policy`
Creates log rotation policies. Suboptions are as follows:
`--policy-name name`. The name of the new log rotation policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Log Rotation Policy that should be created. The value for *type* can be one of `custom, fixed-time`, `size-limit`, or `time-limit`.

`create-workflow-element --type monitor-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type.* The type of Workflow Element that should be created. The value for *type* can be one of `ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, chema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-network-group`
Creates network groups. Suboptions are as follows:
`--group-name` *name*. The name of the new network group.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-network-group-qos-policy`
Creates network group resource limits. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Quality of Service Policy that should be created. The value for *type* can be one of the following `affinity`, `referral`, `request-filtering`, or `resource-limits`.

`create-workflow`
Creates workflows. Suboptions are as follows:
`--workflow-name` *name*. The name of the new workflow. This name will also be used as The value for the `workflow-id` property.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`delete-alert-handler`
Deletes alert handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the alert handler.
`-f,--force`. Ignore nonexistent alert handlers.

`delete-certificate-mapper`
Deletes certificate mappers. Suboptions are as follows:
`--mapper-name` *name*. The name of the certificate mapper.
`-f,--force`. Ignore nonexistent certificate mappers.

`delete-connection-handler`
Deletes connection handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the connection handler.
`-f,--force`. Ignore nonexistent connection handlers.

`delete-debug-target`
Deletes debug targets. Suboptions are as follows:
`--publisher-name` *name*. The name of the debug log publisher.
`--target-name` *name*. The name of the debug target.
`-f,--force`. Ignore nonexistent debug targets.

`delete-extended-operation-handler`
Deletes extended operation handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the extended operation handler.
`-f,--force`. Ignore nonexistent extended operation handlers.

`delete-identity-mapper`
Deletes identity mappers. Suboptions are as follows:
`--mapper-name` *name*. The name of the identity mapper.
`-f,--force`. Ignore nonexistent identity mappers.

`delete-log-publisher`
Deletes log publishers. Suboptions are as follows:

`--publisher-name` *name*. The name of the log publisher.
`-f,--force`. Ignore nonexistent log publishers.

`delete-log-retention-policy`
Deletes Log Retention Policies. Suboptions are as follows:
`--policy-name` *name*. The name of the log retention policy.
`-f,--force`. Ignore nonexistent Log Retention Policies.

`delete-log-rotation-policy`
Deletes log rotation policies. Suboptions are as follows:
`--policy-name` *name*. The name of the log rotation policy.
`-f,--force`. Ignore nonexistent log rotation policies.

`delete-network-group`
Deletes network group. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`-f,--force`. Ignore nonexistent network groups.

`delete-network-group-qos-policy`
Deletes network group quality of service policy. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--policy-type` *name*. The name of the QOS policy.
`-f,--force`. Ignore nonexistent network group resource limits.

`delete-workflow`
Deletes workflow. Suboptions are as follows:
`-f,--force`. Ignore nonexistent workflow.
`--workflow-name` *name*. The name of the workflows.

`delete-workflow-element`
Deletes Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the Workflow Element.
`-f,--force`. Ignore nonexistent workflow elements.

`get-administration-connector-prop`
Shows administration connector properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-alert-handler-prop`
Shows alert handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the alert handler.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-certificate-mapper-prop`
Shows certificate mapper properties. Suboptions are as follows:

`--mapper-name` *name*. The name of the certificate mapper.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-connection-handler-prop`
Shows connection handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the connection handler.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-debug-target-prop`
Shows debug target properties. Suboptions are as follows:
`--publisher-name name`. The name of the debug log publisher.
`--target-name` *name*. The name of the debug target.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-extended-operation-handler-prop`
Shows extended operation handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the extended operation handler.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-global-configuration-prop`
Shows global configuration properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-identity-mapper-prop`
Shows identity mapper properties. Suboptions are as follows:
`--mapper-name` *name*. The name of the identity mapper.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.

`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-log-publisher-prop`
Shows log publisher properties. Suboptions are as follows:
`--publisher-name` *name*. The name of the log publisher.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-log-retention-policy-prop`
Shows log retention policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the log retention policy.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-log-rotation-policy-prop`
Shows log rotation policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the log rotation policy.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-network-group-prop`
Shows network group properties. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-network-group-qos-policy-prop`
Shows network group quality of service policy properties. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--policy-type` *name*. The name of the quality of service policy.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-plugin-root-prop`
Shows plugin root properties.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-root-dse-backend-prop`
Shows root DSE backend properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-work-queue-prop`
Shows work queue properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-workflow-prop`
Shows workflow properties. Suboptions are as follows:
`--workflow-name` *name*. The name of the workflow.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-alert-handlers`
Lists existing alert handlers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-certificate-mappers`
Lists existing certificate mappers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-connection-handlers`
Lists existing connection handlers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-debug-targets`
Lists existing debug targets. Suboptions are as follows:
`--publisher-name` *name*. The name of the Debug Log Publisher.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-extended-operation-handlers`
Lists existing extended operation handlers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-identity-mappers`
Lists existing identity mappers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-log-publishers`
Lists existing log publishers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-log-retention-policies`
Lists existing log retention policies. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-log-rotation-policies`
Lists existing log rotation policies. Suboptions are as follows:

`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-network-group-qos-policies`
Lists existing network group QOS policies. Suboptions are as follows:
`--group-name` *name*. The name of the Network Group.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-network-groups`
Lists existing network groups. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing Workflow Elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflows`
Lists existing workflows. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-administration-connector-prop`
Modifies administration connector properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-alert-handler-prop`
Modifies alert handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the alert handler.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-certificate-mapper-prop`
Modifies certificate mapper properties. Suboptions are as follows:
`--mapper-name` *name.* The name of the certificate mapper.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-connection-handler-prop`
Modifies connection handler properties. Suboptions are as follows:
`--handler-name` *name.* The name of the connection handler.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-debug-target-prop`
Modifies debug target properties. Suboptions are as follows:
`--publisher-name` *name*. The name of the debug log publisher.
`--target-name` *name*. The name of the debug target.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-extended-operation-handler-prop`
Modifies extended operation handler properties. Suboptions are as follows:
`--handler-name` *name.* The name of the extended operation handler.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-global-configuration-prop`
Modifies global configuration properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-identity-mapper-prop`
Modifies identity mapper properties. Suboptions are as follows:
`--mapper-name` *name*. The name of the identity mapper.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-log-publisher-prop`
Modifies log publisher properties. Suboptions are as follows:
`--publisher-name` *name*. The name of the log publisher.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-log-retention-policy-prop`
Modifies log retention policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the log retention policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-log-rotation-policy-prop`
Modifies log rotation policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the log rotation policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-network-group-prop`
Modifies network group properties. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-network-group-qos-policy-prop`
Modifies network group quality of service policy properties. Suboptions are as follows:
`--group-name` *name*. The name of the network group.
`--policy-type` *name*. The name of the QOS policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-plugin-root-prop`
Modifies plugin root properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.

`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-root-dse-backend-prop`
Modifies root DSE back end properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-work-queue-prop`
Modifies work queue properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
Modifies Workflow Element properties. Suboptions are as follows:
--element-name *name*. The name of the Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-prop`
Modifies workflow properties. Suboptions are as follows:
`--workflow-name` *name*. The name of the workflow.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Load Balancing Subcommands

The following subcommands configure load balancing for the proxy server.

`create-load-balancing-algorithm`
This command is supported only for the proxy.
Creates load balancing algorithms. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Load Balancing Algorithm that should be created. The value for *type* can be `failover`, `generic`, `optimal`, `proportional`, `saturation`, or `searchfilter`. The default value is `generic`.

`create-load-balancing-route`
This command is supported only for the proxy.
Creates load balancing routes. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--route-name` *name*. The name of the new load balancing route.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Load Balancing Route that should be created. The value for *type* can be `failover`, `generic`, `optimal`, `proportional`, `saturation`, or `searchfilter`. The default value is `generic`.

`create-workflow-element --type load-balancing`
Creates Workflow Elements. Suboptions are as follows:
--element-name *name*. The name of the new Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `The type of Workflow Element which should be created. The value for TYPE can be one of: ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus,eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing, saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of.`

`delete-load-balancing-algorithm`
Deletes load balancing algorithm. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`-f,--force`. Ignore nonexistent load balancing algorithms.

`delete-load-balancing-route`
Deletes load balancing routes. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--route-name` *name*. The name of the load balancing route.
`-f,--force`. Ignore nonexistent load balancing route.

delete-workflow-element
Deletes Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`-f,--force`. Ignore nonexistent workflow element.

get-load-balancing-algorithm-prop
Shows load balancing algorithm properties. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-load-balancing-route-prop
This command is supported only for the proxy.
Shows load balancing route properties. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--route-name` *name*. The name of the load balancing route.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-load-balancing-algorithm
This command is supported only for the proxy.
Lists existing load balancing algorithm. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-load-balancing-routes
This command is supported only for the proxy.
Lists existing load balancing routes. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-workflow-elements
Lists existing Workflow Elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

set-load-balancing-algorithm-prop
This command is supported only for the proxy.
Modifies load-balancing algorithm properties. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-load-balancing-route-prop
This command is supported only for the proxy.
Modifies load balancing route properties. Suboptions are as follows:
`--element-name` *name*. The name of the load balancing workflow element.
`--route-name` *name*. The name of the load balancing route.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-workflow-element-prop
Modifies Workflow Element properties. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Local Data Source Subcommands

create-account-status-notification-handler
Creates account status notification handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the new account status notification handler.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Account Status Notification Handler that should be created. The value for *type* can be one of `custom`, `error-log`, or `smtp`.

`create-workflow-element --type backup-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus, eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend ,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing,saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of`.

`create-workflow-element --type db-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus, eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend ,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing,saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of`.

`create-entry-cache`
Creates entry caches. Suboptions are as follows:
`--cache-name` *name*. The name of the new Entry Cache.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Entry Cache that should be created. The value for *type* can be one of `custom`, `fifo`, `file-system`, or `soft-reference`.

`create-group-implementation`
This command is not supported for the proxy.
Creates group implementations. Suboptions are as follows:
`--implementation-name` *name*. The name of the new group implementation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Group Implementation that should be created. The value for *type* can be one of `dynamic`, `static`, or `virtual-static`.

`create-workflow-element --type ldif-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus, eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend ,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing,saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of`

`create-local-db-index`
Creates local DB indexes. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the new local DB index, which is also used as the value for the `attribute` property. This specifies the name of the attribute for which the index is to be maintained.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-local-db-vlv-index`
Creates local DB VLV indexes. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the new local DB VLV index, which is also used as the value of the `name` property. This property specifies a unique name for this VLV index.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-workflow-element --`*type*` memory-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus, eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend ,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing,saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of`

`create-workflow-element --`*type*` null-local-backend`
Creates Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the new Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`-t,--type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging,ad-password,backup-local-backend,db-local-backend,distribution,dn-renaming,dynamic-entry-tree,dynamic-groups,eus, eus-alias-resolution,eus-context,fa,flat-tree,fork-join,get-rid-of-duplicate,global-index-local-backend,global-index-replication-changes-local-backend,hide-entries-by-filter,join,kerberos-auth-provider,ldif-local-backend,load-balancing,memory-local-backend,monitor-local-backend ,null-local-backend,pass-through-authentication,plugin,proxy-ldap,rdbms,rdn-changing,saml-xasp,schema-local-backend,transformations,trust-store-local-backend,union,virtual-member-of`

`create-password-generator`
Creates password generators. Suboptions are as follows:
`--generator-name` *name*. The name of the new password generator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of password generator that should be created. The value for *type* can be one of `custom` or `random`.

`create-password-policy`
Creates password Policies. Suboptions are as follows:
`--policy-name` *name*. The name of the new password policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-plugin --`*type*` password-policy-import`
Creates Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the new Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Plugin that should be created. The value for *type* can be one of `dsee-gateway, password-policy-import, referential-integrity, seven-bit-clean, unique-attribute.`

`create-password-storage-scheme`
Creates password storage schemes. Suboptions are as follows:
`--scheme-name name`. The name of the new password storage scheme.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of password Storage scheme that should be created. The value for *type* can be one of `aes`, `base64`, `blowfish`, `clear`, `crypt`, `custom`, `md5`, `rc4`, `salted-md5`, `salted-sha1`, `salted-sha256`, `sha256`, `salted-sha384`, `salted-sha512`, `sha512`, `sha1`, or `triple-des`.

`create-password-validator`
Creates password validators. Suboptions are as follows:
`--validator-name` *name*. The name of the new password validator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`-t,--type` *type*. The type of password validator that should be created. The value for *type* can be one of `attribute-value`, `character-set`, `custom`, `dictionary`, `length-based`, `repeated-characters`, `similarity-based`, or `unique-characters`.

`create-plugin --type referential-integrity`
Creates Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the new Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Plugin that should be created. The value for *type* can be one of `dsee-gateway`, `password-policy-import`, `referential-integrity`,`seven-bit-clean`,`unique-attribute`.

`create-plugin --type seven-bit-clean`
Creates Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the new Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Plugin that should be created. The value for *type* can be one of `dsee-gateway`, `password-policy-import`, `referential-integrity`,`seven-bit-clean`,`unique-attribute`.

`create-plugin --type unique-attribute`
Creates Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the new Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Plugin that should be created. The value for *type* can be one of `dsee-gateway`, `password-policy-import`, `referential-integrity`,`seven-bit-clean`,`unique-attribute`.

`create-virtual-attribute`
This command is not supported for the proxy.
Creates virtual attributes. Suboptions are as follows:
`--name` *name*. The name of the new virtual attribute.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Virtual Attribute that should be created. The value for *type* can be one of `collective-attribute-subentries`, `custom`, `entry-dn`,`entry-uuid`, `governing-structure-rule`, `has-subordinates`, `is-member-of`, `member`, `nsuniqueid`, `num-subordinates`, `orclguid`, `password-policy-subentry`, `proximity`, `structural-object-class`, `subschema-subentry`, `user-defined`.

`delete-account-status-notification-handler`
Deletes account status notification handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the account status notification handler.
`-f,--force`. Ignore nonexistent account status notification handlers.

`delete-entry-cache`
Deletes entry caches. Suboptions are as follows:
`--cache-name` *name*. The name of the Entry Cache.

`-f,--force`. Ignore nonexistent entry cache.

`delete-group-implementation`
This command is not supported for the proxy.
Deletes group implementations. Suboptions are as follows:
`--implementation-name` *name*. The name of the group implementation.
`-f,--force`. Ignore nonexistent group implementations.

`delete-local-db-index`
Deletes local DB indexes. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the local DB index.
`-f,--force`. Ignore nonexistent local DB indexes.

`delete-local-db-vlv-index`
Deletes local DB VLV indexes. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the local DB VLV index.
`-f,--force`. Ignore nonexistent local DB VLV indexes.

`delete-password-generator`
Deletes password generators. Suboptions are as follows:
`--generator-name` *name*. The name of the password generator.
`-f,--force`. Ignore nonexistent password generators.

`delete-password-policy`
Deletes password policies. Suboptions are as follows:
`--policy-name` *name*. The name of the password policy.
`-f,--force`. Ignore nonexistent password policies.

`delete-password-storage-scheme`
Deletes password storage schemes. Suboptions are as follows:
`--scheme-name` *name*. The name of the password storage scheme.
`-f,--force`. Ignore nonexistent password storage schemes.

`delete-password-validator`
Deletes password validators. Suboptions are as follows:
`--validator-name` *name*. The name of the password validator.
`-f,--force`. Ignore nonexistent password validators.

`delete-plugin`
Deletes Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the Plugin.
`-f,--force`. Ignore nonexistent Plugins.

`delete-virtual-attribute`
This command is not supported for the proxy.
Deletes virtual attributes. Suboptions are as follows:
`--name name`. The name of the virtual attribute.
`-f,--force`. Ignore nonexistent virtual attributes.

`delete-workflow-element`
Deletes Workflow Elements. Suboptions are as follows:
`--element-name` *name*. The name of the Workflow Element.
`-f,--force`. Ignore nonexistent Workflow Elements.

get-account-status-notification-handler-prop
Shows account status notification handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the account status notification handler.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-entry-cache-prop
Shows entry cache properties. Suboptions are as follows:
`--cache-name` *name*. The name of the entry cache.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-group-implementation-prop
This command is not supported for the proxy.
Shows group implementation properties. Suboptions are as follows:
`--implementation-name` *name*. The name of the group implementation.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-local-db-index-prop
Shows local DB index properties. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the local DB index.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-local-db-vlv-index-prop
Shows the local DB VLV index properties. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end.
`--index-name` *name*. The name of the local DB VLV index.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-password-generator-prop
Shows password generator properties. Suboptions are as follows:
`--generator-name` *name*. The name of the password generator.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-password-policy-prop
Shows password policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the password policy.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-password-storage-scheme-prop
Shows password storage scheme properties. Suboptions are as follows:
`--scheme-name` *name*. The name of the password storage scheme.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-password-validator-prop
Shows password validator properties. Suboptions are as follows:
`--validator-name` *name*. The name of the password validator.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-virtual-attribute-prop
This command is not supported for the proxy.
Shows virtual attribute properties. Suboptions are as follows:
`--name` *name*. The name of the virtual attribute.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-account-status-notification-handlers
Lists existing account status notification handlers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.

`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-entry-caches`
Lists existing entry caches. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-group-implementations`
This command is not supported for the proxy.
Lists existing group implementations. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-local-db-indexes`
Lists existing local DB indexes. Suboptions are as follows:
`--element-name` *name*. The name of the DB local backend Workflow Element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-local-db-vlv-indexes`
Lists existing local DB VLV indexes. Suboptions are as follows:
`--element-name` *name*. The name of the DB local backend Workflow Element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-generators`
Lists existing password generators. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-policies`
Lists existing password policies. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-storage-schemes`
Lists existing password storage schemes. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-validators`
Lists existing password validators. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-plugins`
Lists existing Plugins. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-virtual-attributes`
This command is not supported for the proxy.
Lists existing virtual attributes. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing Workflow Elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-account-status-notification-handler-prop`
Modifies account status notification handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the account status notification handler.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.

`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-entry-cache-prop`
Modifies Entry Cache properties. Suboptions are as follows:
`--cache-name` *name*. The name of the Entry Cache.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-group-implementation-prop`
This command is not supported for the proxy.
Modifies group implementation properties. Suboptions are as follows:
`--implementation-name` *name.* The name of the group implementation.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-local-db-index-prop`
Modifies local DB Index properties. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the local DB Index.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-local-db-vlv-index-prop`
Modifies local DB VLV Index properties. Suboptions are as follows:
`--element-name` *name*. The name of the local DB back end workflow element.
`--index-name` *name*. The name of the local DB VLV Index.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.

`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-password-generator-prop`
Modifies password generator properties. Suboptions are as follows:
`--generator-name` *name*. The name of the password generator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-password-policy-prop`
Modifies password policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the password policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-password-storage-scheme-prop`
Modifies password storage scheme properties. Suboptions are as follows:
`--scheme-name` *name*. The name of the password storage scheme.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-password-validator-prop`
Modifies password validator properties. Suboptions are as follows:
`--validator-name` *name*. The name of the password validator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.

`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-plugin-prop`
Modifies Plugin properties. Suboptions are as follows:
`--plugin-name` *name*. The name of the Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-virtual-attribute-prop`
This command is not supported for the proxy.
Modifies virtual attribute properties. Suboptions are as follows:
`--name` *name.* The name of the virtual attribute.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
Modifies Workflow Element properties. Suboptions are as follows:
`--element-name` *name.* The name of the Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Integration Subcommands

This section describes the subcommands for various workflow operations.

`create-workflow-element --type ad-paging`
This command creates Ad Paging Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the new Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend,`

distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type ad-password
This command creates password Workflow Elements. Suboptions are as follows:
--element-name *name.* The name of the Workflow Element.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type eus-context
This command creates Eus Context Workflow Elements. Suboptions are as follows:
--element-name *name.* The name of the new Workflow Element.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type eus
This command creates Eus Workflow Elements. Suboptions are as follows:
--element-name *name.* The name of the new Workflow Element.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-

balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type fa

This command creates Fa Workflow Elements. Suboptions are as follows:

--element-name *name.* The name of the new Workflow Element.

--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type kerberos-auth-provider

This command creates Kerberos Auth Provider Workflow Elements. Suboptions are as follows:

--element-name *name.* The name of the new Workflow Element.

--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type pass-through-authentication

This command creates Pass Through Authentication Workflow Elements. Suboptions are as follows:

--element-name *name.* The name of the new Workflow Element.

--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp,

schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

create-workflow-element --type plugin
This command creates Plugin Workflow Elements. Suboptions are as follows:
--element-name *name.* The name of the new Workflow Element.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
-t, --type *type*. The type of Workflow Element that should be created. The value for *type* can be one of ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of

delete-workflow-element
This command deletes Workflow Elements. Suboptions are as follows:
--element-name *name.* The name of the Workflow Element.
-f, --force. Ignore nonexistent Workflow Elements.

list-workflow-elements
Lists existing workflow elements. Suboptions are as follows:
--property *property*. The name of a property to be displayed.
-z,--unit-size *unit*. Displays size data using the specified unit. The value for *unit* can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes).
-m,--unit-time *unit*. Displays time data using the specified unit. The value for *unit* can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks).

set-workflow-element-prop
Modifies workflow element properties. Suboptions are as follows:
--element-name *name*. The name of the workflow element.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
--reset *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
--add *property*:*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
--remove *property*:*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Remote Data Source Subcommands

This section describes subcommands for various remote data source operations.

create-extension --type ldap-server
This command creates LDAP Server Extensions. Suboptions are as follows:
--extension-name *name.* The name of the new extension.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Extension that should be created. The value for *type* can be one of `global-index-catalog, global-index-catalogs-shared-cache,ldap-server.`

`create-workflow-element --type proxy-ldap`
This command creates Proxy LDAP Workflow Elements. Suboptions are as follows:
`--element-name` *name.* The name of the new workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Workflow Element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-contex, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations,trust-store-local-backend, union, virtual-member-of`

`delete-extension`
Deletes extension. Suboptions are as follows:
`--extension-name` *name*. The name of the extension.
`-f,--force`. Ignore nonexistent extensions.

`delete-workflow-element`
Deletes workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`-f,--force`. Ignore nonexistent workflow elements.

`list-extensions`
Lists existing extensions. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing workflow elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-extension-prop`
This command modifies Extension properties. Suboptions are as follows:
`--extension-name` *name*. The name of the Extension.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
This command modifies Workflow Element properties. Suboptions are as follows:
`--element-name` *name.* The name of the Workflow Element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Replication Subcommands

This section describes subcommands for various replication operations.

`create-plugin --`*type* `dsee-gateway`
Creates Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t, --type` *type*. The type of Plugin that should be created. The value for *type* can be one of `dsee-gateway,password-policy-import,referential-integrity, seven-bit-clean,unique-attribute.`

`create-gateway-domain`
Creates gateway domains. Suboptions are as follows:
`--plugin-name` *name*. The name of the DSEE gateway plugin.
`--domain-name` *name*. The name of the gateway domain.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-replication-domain`
Creates replication domains. Suboptions are as follows:
`--provider-name` *name*. The name of the multi-master synchronization provider.
`--domain-name` *name*. The name of the new replication domain.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-replication-server`
Creates replication servers. Suboptions are as follows:
`--provider-name` *name*. The name of the multi-master synchronization provider.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-synchronization-provider`
Creates synchronization providers. Suboptions are as follows:
`--provider-name` *name*. The name of the new synchronization provider.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Synchronization Provider that should be created. The value for *type* can be one of `custom`,`replication`.

`delete-gateway-domain`
Deletes gateway domains. Suboptions are as follows:
`--plugin-name` *name*. The name of the DSEE gateway plugin.
`--domain-name` *name*. The name of the gateway domain.
-f, --force. Ignore nonexistent Gateway Domains.

`delete-plugin`
Deletes Plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the Plugin.
`-f,--force`. Ignore nonexistent Plugin.

`delete-replication-domain`
Deletes replication domains. Suboptions are as follows:
`--provider-name` *name*. The name of the synchronization provider.
`--domain-name` *name*. The name of the replication domain.
`-f,--force`. Ignore nonexistent replication domains.

`delete-replication-server`
Deletes replication servers. Suboptions are as follows:
`--provider-name` *name*. The name of the synchronization provider.
`-f,--force`. Ignore nonexistent replication servers.

`delete-synchronization-provider`
Deletes synchronization providers. Suboptions are as follows:
`--provider-name` *name*. The name of the synchronization provider.
`-f,--force`. Ignore nonexistent synchronization providers.

`get-external-changelog-domain-prop`
Shows External Changelog Domain properties. Suboptions are as follows:
`--provider-name` *name*. The name of the Replication Synchronization Provider.
`--domain-name` *name*. The name of the Replication Domain.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-gateway-domain-prop`
Shows gateway domain properties.
`--plugin-name` *name*. The name of the DSEE gateway plugin.
`--domain-name` *name*. The name of the gateway domain.

`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-replication-domain-prop`
Shows replication domain properties. Suboptions are as follows:
`--provider-name` *name*. The name of the multi-master synchronization provider.
`--domain-name` *name*. The name of the replication domain.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-replication-server-prop`
Shows replication server properties. Suboptions are as follows:
`--provider-name name`. The name of the multi-master synchronization provider.
`--property` *property*. The name of a property to be displayed.
`-E, --record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-synchronization-provider-prop`
Shows synchronization provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the synchronization provider.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-plugins`
Lists existing Plugins. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-gateway-domains`
Lists existing gateway domains. Suboptions are as follows.
`--plugin-name` *name*. The name of the DSEE Gateway Plugin.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-replication-domains
Lists existing replication domains. Suboptions are as follows:
`--provider-name` *name*. The name of the replication synchronization provider.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-replication-server
Lists existing replication server. Suboptions are as follows:
`--provider-name` *name*. The name of the replication synchronization provider.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-synchronization-providers
Lists existing synchronization providers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

set-external-changelog-domain-prop
Modifies External Changelog Domain properties. Suboptions are as follows:
`--provider-name` *name*. The name of the Replication Synchronization Provider.
`--domain-name` *name*. The name of the Replication Domain.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-gateway-domain-prop
Modifies gateway domain properties. Suboptions are as follows:
`--plugin-name` *name*. The name of the DSEE Gateway Plugin.
`--domain-name` *name*. The name of the gateway domain.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-plugin-prop
Modifies Plugin properties. Suboptions are as follows:
`--plugin-name` *name*. The name of the Plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-replication-domain-prop
Modifies replication domain properties. Suboptions are as follows:
`--provider-name` *name*. The name of the replication synchronization provider.
`--domain-name` *name*. The name of the replication domain.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-replication-server-prop
Modifies replication server properties. Suboptions are as follows:
`--provider-name` *name*. The name of the replication synchronization provider.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-synchronization-provider-prop
Modifies synchronization provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the synchronization provider.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Schema Subcommands

This section describes subcommands for various schema operations.

create-attribute-syntax
This command is not supported for the proxy.
Creates attribute syntaxes. Suboptions are as follows:
`--syntax-name` *name*. The name of the new attribute syntax.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Attribute Syntax that should be created. The value for *type* can be one of `attribute-type-description`, `directory-string`, `generic`, or `telephone-number`.

create-matching-rule
This command is not supported for the proxy.
Creates matching rules. Suboptions are as follows:
`--rule-name` *name*. The name of the new matching rule.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Matching Rule that should be created. The value for *type* can be one of `collation` or `generic`.

delete-attribute-syntax
This command is not supported for the proxy.
Deletes attribute syntaxes. Suboptions are as follows:
`--syntax-name` *name*. The name of the attribute syntax.
`-f,--force`. Ignore nonexistent attribute syntaxes.

delete-matching-rule
This command is not supported for the proxy.
Deletes matching rules. Suboptions are as follows:
`--rule-name` *name*. The name of the matching rule.
`-f,--force`. Ignore nonexistent matching rules.

get-attribute-syntax-prop
This command is not supported for the proxy.
Shows attribute syntax properties. Suboptions are as follows:
`--syntax-name` *name*. The name of the attribute syntax.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-matching-rule-prop
This command is not supported for the proxy.
Shows matching rule properties. Suboptions are as follows:
`--rule-name` *name*. The name of the matching rule.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-attribute-syntaxes
This command is not supported for the proxy.
Lists existing attribute syntaxes. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-matching-rules
This command is not supported for the proxy.
Lists existing matching rules. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

set-attribute-syntax-prop
This command is not supported for the proxy.
Modifies attribute syntax properties. Suboptions are as follows:
`--syntax-name` *name*. The name of the attribute syntax.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-matching-rule-prop
This command is not supported for the proxy.
Modifies matching rule properties. Suboptions are as follows:
`--rule-name` *name*. The name of the matching rule.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Security Subcommands

create-access-control-group
Creates access control groups.
`--group-name` *name*. The name of the new access control group.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

create-key-manager-provider
Creates key manager providers. Suboptions are as follows:
`--provider-name` *name*. The name of the new key manager provider.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type type`. The type of key manager provider that should be created. The value for *type* can be one of `file-based`, `custom`, or `pkcs11`.
PKCS#11 is not supported for a proxy server instance.

create-key-manager-provider-key-pin
Creates key manager provider key pins. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--pin-name` *name*. The name of the new key manager provider key pin which will also be used as the value of the "ssl-cert-nickname" property: Specifies the nickname of the certificate this key manager provider key pin applies to.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

create-password-generator
Creates password generators. Suboptions are as follows:
`--generator-name` *name*. The name of the new password generator.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type type`. The type of password generator which should be created. The value for *type* can be one of: `custom`, or `random`.

create-password-policy
Creates password policies. Suboptions are as follows:
`--policy-name` *name*. The name of the new password policy.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

create-plugin --type password-policy-import
Creates password policy import plugins. Suboptions are as follows:
`--plugin-name` *name*. The name of the new plugin.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type type`. The type of plugin which should be created. The value for *type* can be one of: `dsee-gateway, last-mod, password-policy-import, referential-integrity, seven-bit-clean, unique-attribute`.

create-password-storage-scheme
Creates password storage schemes. Suboptions are as follows:
`--scheme-name` *name*. The name of the new password storage scheme.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type type`. The type of password storage scheme which should be created. The value for *type* can be one of: `aes, base64, blowfish, clear, crypt, custom, euspbkdf2sha512, md5, pbkdf2hmacsha1, pbkdf2hmacsha256, pbkdf2hmacsha512, rc4, salted-md5,`

`salted-sha1, salted-sha256, salted-sha384, salted-sha512, sha1, sha256, sha512, triple-des, user-defined`.

`create-password-validator`
Creates password validators. Suboptions are as follows:
`--validator-name` *name*. The name of the new password validator.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type type`. The type of password validator which should be created. The value for *type* can be one of: `character-set, custom, dictionary, length-based, repeated-characters, similarity-based, unique-characters`.

`create-sasl-mechanism-handler`
This command is not supported for the proxy.
Creates SASL mechanism handlers. Suboptions are as follows:
`--handler-name` *name*. The name of the new SASL mechanism handler.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of SASL Mechanism Handler that should be created. The value for *type* can be one of `anonymous`, `cram-md5`, `digest-md5`, `external`, `custom`, `gssapi`, or `plain`.

`create-trust-manager-provider`
Creates trust manager providers. Suboptions are as follows:
`--provider-name` *name*. The name of the new trust manager provider.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of trust manager provider that should be created. The value for *type* can be one of `blind`, `file-based`, or `custom`.

`create-trust-store-key-pin`
Creates trust store key pins. Suboptions are as follows:
`--element-name` *name*. The name of the trust store local backend workflow element.
`--pin-name` *string*. The name of the new trust store key pin which will also be used as the value of the "ssl-cert-nickname" property: Specifies the nickname of the certificate this trust store key pin applies to.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-workflow-element --`*type* `trust-store-local-backend`
Creates workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, backup-local-backend, db-local-backend,distribution, dn-renaming, eus, eus-context, fa, global-index-local-backend, global-index-replication-changes-local-backend, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdn-changing, transformations, trust-store-local-backend`.

delete-access-control-group
Deletes access control groups. Suboptions are as follows:
`--group-name` *name*. The name of the access control group.
`-f,--force`. Ignore nonexistent access control groups.

delete-key-manager-provider
Deletes key manager providers. Suboptions are as follows:
`--provider-name` *name.* The name of the key manager provider.
`-f,--force`. Ignore nonexistent key manager providers.

delete-key-manager-provider-key-pin
Deletes key manager provider key pins. Suboptions are as follows:
`--provider-name` *name.* The name of the key manager provider.
`--pin-name` *name.* The name of the key manager provider key pin.
`-f,--force`. Ignore nonexistent key manager provider key pins.

delete-password-generator
Deletes password generators. Suboptions are as follows:
`--generator-name` *name.* The name of the password generator.
`-f,--force`. Ignore nonexistent password generators.

delete-password-policy
Deletes password Policies. Suboptions are as follows:
`--policy-name` *name.* The name of the password policy.
`-f,--force`. Ignore nonexistent password Policies.

delete-password-storage-scheme
Deletes password storage schemes. Suboptions are as follows:
`--scheme-name` *name.* The name of the password storage scheme.
`-f,--force`. Ignore nonexistent password storage schemes.

delete-password-validator
Deletes password validators. Suboptions are as follows:
`--validator-name` *name.* The name of the password validator.
`-f,--force`. Ignore nonexistent password validators.

delete-plugin
Deletes plugins. Suboptions are as follows:
`--plugin-name` *name.* The name of the plugin.
`-f,--force`. Ignore nonexistent plugins.

delete-sasl-mechanism-handler
This command is not supported for the proxy.
Deletes SASL mechanism handlers. Suboptions are as follows:
--handler-name *name*. The name of the SASL mechanism handler.
`-f,--force`. Ignore nonexistent SASL mechanism handlers.

delete-trust-manager-provider
Deletes trust manager providers. Suboptions are as follows:
`--provider-name` *name*. The name of the trust manager provider.
`-f,--force`. Ignore nonexistent trust manager providers.

delete-trust-store-key-pin
Deletes trust manager providers. Suboptions are as follows:
`--element-name` *name*. The name of the trust store local backend workflow element.

`--pin-name` *name*. The name of the trust store key pin.
`-f,--force`. Ignore nonexistent trust store key pins.

`delete-workflow-element`
Deletes workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`-f,--force`. Ignore nonexistent workflow elements.

`get-access-control-group-prop`
Shows access control group properties. Suboptions are as follows:
`--group-name` *name*. The name of the access control group.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-access-control-handler-prop`
Shows access control handler properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-crypto-manager-prop`
Show crypto manager properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-data-encryption-prop`
Shows data encryption properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-key-manager-provider-key-pin-prop`
Shows key manager provider key pin properties. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--pin-name` *name*. The name of the key manager provider key pin.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-key-manager-provider-prop`
Shows key manager provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-password-generator-prop`
Shows password generator properties. Suboptions are as follows:
`--generator-name` *name*. The name of the password generator.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-password-policy-prop`
Shows password policy properties. Suboptions are as follows:
`--policy-name` *name*. The name of the password policy.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-password-storage-scheme-prop`
Shows password storage scheme properties. Suboptions are as follows:
`--scheme-name` *name*. The name of the password storage scheme.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`get-password-validator-prop`
Shows password validator properties. Suboptions are as follows:
`--validator-name` *name*. The name of the password validator.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-root-dn-prop
Shows root DN properties. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-sasl-mechanism-handler-prop
Shows SASL mechanism handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the SASL mechanism handler.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-trust-manager-provider-prop
Shows trust manager provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the trust manager provider.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-trust-store-key-pin-prop
Shows trust store key pin properties. Suboptions are as follows:
`--element-name` *name*. The name of the trust store local backend workflow element.
`--pin-name` *name*. The name of the trust store key pin.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-access-control-groups
Lists existing access control groups. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-key-manager-provider-key-pins
Lists existing key manager provider key pins. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-key-manager-providers`
Lists existing key manager providers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-generators`
Lists existing password generators. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-policies`
Lists existing password Policies. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-storage-schemes`
Lists existing password storage schemes. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-password-validators`
Lists existing password validators. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-plugins`
Lists existing plugins. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-sasl-mechanism-handlers`
This command is not supported for the proxy.
Lists existing SASL mechanism handlers. Suboptions are as follows:

`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-trust-manager-providers`
Lists existing trust manager providers. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-trust-store-key-pins`
Lists existing trust store key pins. Suboptions are as follows:
`--element-name` *name*. The name of the trust store local backend workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing workflow elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-access-control-group-prop`
Modifies access control group properties. Suboptions are as follows:
`--group-name` *name*. The name of the access control group.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-access-control-handler-prop`
Modifies access control handler properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.

`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-crypto-manager-prop`
Modifies crypto manager properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-data-encryption-prop`
Modifies data encryption properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-key-manager-provider-key-pin-prop`
Modifies key manager provider key pin properties. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--pin-name` *name*. The name of the key manager provider key pin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-key-manager-provider-prop`
Modifies key manager provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the key manager provider.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-password-generator-prop
Modifies password generator properties. Suboptions are as follows:
`--element-name` *name*. The name of the password generator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-password-policy-prop
Modifies password policy properties. Suboptions are as follows:
`--element-name` *name*. The name of the password policy.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-password-storage-scheme-prop
Modifies password storage scheme properties. Suboptions are as follows:
`--scheme-name` *name*. The name of the password storage scheme.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-password-validator-prop
Modifies password validator properties. Suboptions are as follows:
`--validator-name` *name*. The name of the password validator.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

set-plugin-prop
Modifies plugin properties. Suboptions are as follows:

`--plugin-name` *name*. The name of the plugin.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-root-dn-prop`
Modifies root DN properties. Suboptions are as follows:
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-sasl-mechanism-handler-prop`
This command is not supported for the proxy.
Modifies SASL mechanism handler properties. Suboptions are as follows:
`--handler-name` *name*. The name of the SASL mechanism handler.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-trust-manager-provider-prop`
Modifies trust manager provider properties. Suboptions are as follows:
`--provider-name` *name*. The name of the trust manager provider.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-trust-store-key-pin-prop`
Modifies trust store key pin properties. Suboptions are as follows:
`--element-name` *name*. The name of the trust store local backend workflow element.
`--pin-name` *name*. The name of the trust store key pin.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
Modifies workflow element properties. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Virtualization Subcommands

This section describes subcommands for virtualization.

`create-transformation --type add-inbound-attribute`
Creates add inbound attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

`create-transformation --type add-outbound-attribute`
Creates add outbound attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

`create-workflow-element --type dn-renaming`
Creates DN renaming workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.

`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-workflow-element --type dynamic-entry-tree`
Creates dynamic entry tree workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-workflow-element --type dynamic-groups`
Creates dynamic groups workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-transformation --type filter-inbound-attribute`
Creates filter inbound attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

`create-transformation --type filter-outbound-attribute`
Creates filter outbound attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for type can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

`create-workflow-element --type flat-tree`
Creates flat tree workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-workflow-element --type fork-join`
Creates fork join workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-workflow-element --type get-rid-of-duplicate`
Creates get rid of duplicate workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.

`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-workflow-element --type hide-entries-by-filter`
Creates hide entries by filter workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-join-participant`
Creates join participants. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--participant-name` *name*. The name of the new join participant.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-workflow-element --type join`
Creates join workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

create-transformation --type map-attribute
Creates map attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

create-transformation --type map-object-class
Creates map object class transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`. For more information about each transformation, see Configuring Transformation Using `dsconfig`.

create-primary-fork-join-participant
Creates primary fork join participants. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

create-workflow-element --type rdn-changing
Creates RDN changing workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

create-workflow-element --type saml-xasp
Creates SAML XASP workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

`create-secondary-fork-join-participant`
Creates secondary fork join participants. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`create-transformation --type tokenize-attribute`
Creates tokenize attribute transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`.

`create-transformation`
Creates transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the new transformation.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of Transformation that should be created. The value for *type* can be one of `add-inbound-attribute, add-outbound-attribute, filter-inbound-attribute, filter-outbound-attribute, map-attribute, map-object-class, tokenize-attribute`.

`create-workflow-element --type transformations`
Creates transformations workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging, ad-password, backup-local-backend, db-local-backend, distribution, dn-renaming, dynamic-entry-tree, dynamic-groups, eus, eus-alias-resolution, eus-context, fa, flat-tree, fork-join, get-rid-of-duplicate, global-index-local-backend, global-index-replication-changes-local-backend, hide-entries-by-filter, join, kerberos-auth-provider, ldif-local-backend, load-balancing, memory-local-backend, monitor-local-backend, null-local-backend, pass-through-authentication, plugin, proxy-ldap, rdbms, rdn-changing, saml-xasp, schema-local-backend, transformations, trust-store-local-backend, union, virtual-member-of`.

create-workflow-element --type virtual-member-of
Creates virtual member of workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the new workflow element.
`--set` *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`-t,--type` *type*. The type of workflow element that should be created. The value for *type* can be one of `ad-paging`, `ad-password`, `backup-local-backend`, `db-local-backend`, `distribution`, `dn-renaming`, `dynamic-entry-tree`, `dynamic-groups`, `eus`, `eus-alias-resolution`, `eus-context`, `fa`, `flat-tree`, `fork-join`, `get-rid-of-duplicate`, `global-index-local-backend`, `global-index-replication-changes-local-backend`, `hide-entries-by-filter`, `join`, `kerberos-auth-provider`, `ldif-local-backend`, `load-balancing`, `memory-local-backend`, `monitor-local-backend`, `null-local-backend`, `pass-through-authentication`, `plugin`, `proxy-ldap`, `rdbms`, `rdn-changing`, `saml-xasp`, `schema-local-backend`, `transformations`, `trust-store-local-backend`, `union`, `virtual-member-of`.

delete-join-participant
Deletes join participants. Suboptions are as follows:
`--element-name` *name*. The name of the Join workflow element.
`--participant-name` *name*. The name of the join participant.
`-f, --force`. Ignore nonexistent join participants.

delete-primary-fork-join-participant
Deletes primary fork join participants. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`-f, --force`. Ignore nonexistent primary fork join participants.

delete-secondary-fork-join-participant
Deletes secondary fork join participants. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`-f, --force`. Ignore nonexistent secondary fork join participants.

delete-transformation
Deletes transformations. Suboptions are as follows:
`--transformation-name` *name*. The name of the transformation.
`-f, --force`. Ignore nonexistent transformation.

delete-workflow-element
Deletes workflow elements. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`-f, --force`. Ignore nonexistent workflow elements.

get-join-participant-prop
Shows join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the join workflow element.
`--participant-name` *name*. The name of the join participant.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-primary-fork-join-participant-prop
Shows primary fork join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-secondary-fork-join-participant-prop
Shows secondary fork join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

get-transformation-prop
Shows transformation properties. Suboptions are as follows:
`--transformation-name` *name*. The name of the transformation element.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-join-participants
Lists existing join participants. Suboptions are as follows:
`--element-name` *name*. The name of the join workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-primary-fork-join-participant
Lists existing primary fork join participant. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

list-secondary-fork-join-participant
Lists existing secondary fork join participant. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).

`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-transformations`
Lists existing transformations. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`list-workflow-elements`
Lists existing workflow elements. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed.
`-z,--unit-size` *unit*. Displays size data using the specified unit. The value for *unit* can be one of `b`, `kb`, `mb`, `gb`, or `tb` (bytes, kilobytes, megabytes, gigabytes, or terabytes).
`-m,--unit-time` *unit*. Displays time data using the specified unit. The value for *unit* can be one of `ms`, `s`, `m`, `h`, `d`, or `w` (milliseconds, seconds, minutes, hours, days, or weeks).

`set-join-participant-prop`
Modifies join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the join workflow element.
`--participant-name` *name*. The name of the join participant.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-primary-fork-join-participant-prop`
Modifies primary fork join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-secondary-fork-join-participant-prop`
Modifies secondary fork join participant properties. Suboptions are as follows:
`--element-name` *name*. The name of the fork join workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.

`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-transformation-prop`
Modifies transformation properties. Suboptions are as follows:
`--transformation-name` *name*. The name of the transformation element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

`set-workflow-element-prop`
Modifies workflow element properties. Suboptions are as follows:
`--element-name` *name*. The name of the workflow element.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.
`--reset` *property*. Resets a property back to its default values, where *property* is the name of the property to be reset.
`--add` *property*`:`*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
`--remove` *property*`:`*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed.

Options

The `dsconfig` command accepts an option in either its short form (for example, `-h hostname`) or its long form equivalent (for example, `--hostname hostname`).

`--advanced`
Allows the configuration of advanced components and properties.

`--showKeystorePassword`
Retrieves the keystore or truststore password in the text format on the terminal.

LDAP Connection Options

The `dsconfig` command contacts the directory server over SSL through the administration connector (described in Managing Administration Traffic to the Server). These connection options are used to contact the directory server.

`-D, --bindDN bindDN`
Use the bind DN to bind the server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.
SASL is not supported for a proxy server instance.

-h, --hostname hostname
Contact the server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

-j, --bindPasswordFile filename
Use the bind password in the specified file when authenticating to the server.

-K, --keyStorePath path
Use the client keystore certificate in the specified path.

-N, --certNickname nickname
Use the nickname of certificate for SSL client authentication.

-o, --saslOption name=value
Use the specified options for SASL authentication.
SASL is not supported for a proxy server instance.

-p, --port port
Contact the server at the specified administration port. If this option is not provided, the administration port of the local configuration is used.

-P, --trustStorePath path
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

-u, --keyStorePasswordFile filename
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

-U, --trustStorePasswordFile filename
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

-X, --trustAll
Trust all server SSL certificates that the server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate. If the client and the server run in the same instance, there is no certificate interaction.

--connectTimeout {timeout}
This is used to specify the maximum length of time (in milliseconds) that can be taken to establish a connection. Use `0` to specify no time out. The default value is `30000`.

Command Input/Output Options

--commandFilePath path
Specify the full path to the file, where the equivalent non-interactive commands will be written when this command is run in interactive mode.

--displayCommand
Display the equivalent non-interactive option in the standard output when this command is run in interactive mode.

-F, --batchFilePath batchFilePath
Specifies the path to a file that contains a set of `dsconfig` commands to be executed. This option supports line splitting, backslash ('\'), quotes (") escaped quotes (\") inside a quoted string, and hash for comments ('#').

-n, --no-prompt
Use non-interactive mode. If some data in the command is missing, you are not prompted and the command will fail.

--noPropertiesFile
Indicate that the command will not use a properties file to get the default command-line options.

--sortMenuItems
Allows to sort the menu items if the interactive mode is used. The order is the user locale alphabetic order.

--propertiesFilePath path
Specify the path to the properties file that contains the default command-line options.

-Q, --quiet
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

-s, --script-friendly
Run in "script friendly" mode. Display the output in a format that can be easily parsed by a script.

-v, --verbose
Run in verbose mode, displaying diagnostics on standard output.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

-V, --version
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `dsconfig` command. For additional `dsconfig` examples, see Managing the Server Configuration Using `dsconfig`.

Viewing the Global Help Subcommands and Global Options

The following command displays the available global help subcommands and global options for the server:

```
$ dsconfig --help
```

Viewing a Component's Subcommand Help Information

The following command displays subcommands relating to authentication and authorization:

```
$ dsconfig --help-security
```

Viewing Help on an Individual Subcommand

The following command displays the help information for the `set-distribution-partition-prop` subcommand:

```
$ dsconfig set-distribution-partition-prop --help
```

Displaying a Component's Properties

The following command displays the properties for `local-db-index`. If `-t` is not specified, the command displays the properties for all components.

```
$ dsconfig list-properties -c local-db-index
Option Types:

 r -- Property value(s) are readable
 w -- Property value(s) are writable
 m -- The property is mandatory
 s -- The property is single-valued
 a -- Administrative action is required for changes to take effect

Component       Type     Property                         Options  Syntax
-------------------------------------------------------------------------------
local-db-index  generic  attribute                        r-ms-    OID
local-db-index  generic  index-entry-limit                rw-sa    INTEGER
local-db-index  generic  index-extensible-matching-rule   rw--a    LOCALE | OID
local-db-index  generic  index-type                       rwm-a    TYPE
```

The following command displays the properties for `crypto-manager`.

```
$ dsconfig list-properties -c crypto-manager
Option Types:

 r -- Property value(s) are readable
 w -- Property value(s) are writable
 m -- The property is mandatory
 s -- The property is single-valued
 a -- Administrative action is required for changes to take effect

Component       Type     Property                     Options  Syntax
-----------------------------------------------------------------------
crypto-manager  generic  key-wrapping-transformation  rw-s-    STRING
crypto-manager  generic  ssl-cert-nickname            rw-sa    STRING
crypto-manager  generic  ssl-cipher-suite             rw---    STRING
crypto-manager  generic  ssl-encryption               rw-s-    BOOLEAN
crypto-manager  generic  ssl-protocol                 rw---    STRING
```

Parameters Supported by the `-F, --batchFilePath` subcommand

The following example describes the various parameters supported by the `-F, --batchFilePath` subcommand.

Executing the `-F, --batchFilePath` subcommand using the line splitting approach. The file `/tmp/batch` contains the following set of commands:

```
create-workflow-element \
--type db-local-backend \
--set base-dn:cn=myexample,cn=com \
--set enabled:true \
--element-name myBackend
```

Running the `-F, --batchFilePath` subcommand.

```
dsconfig -X -j /path/pwd-file -F /tmp/batch -n
```

Executing the `-F, --batchFilePath` subcommand using quotes (") and escaped quotes (\") inside a quoted string. The file /tmp/batch contains the following set of commands:

```
set-access-control-handler-prop \
--add global-aci:"(targetattr != \"description || mail\") \
(version 3.0; acl \"Allow self entry modification except for \
description and mail attributes\"; allow (write)userdn =\"ldap:///self\";) "
```

Running the `-F, --batchFilePath` subcommand.

```
dsconfig -X  -j /path/pwd-file -F /tmp/batch -n
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 or greater indicates that an error occurred during processing.

How to Use a Properties File

The server supports the use of a *properties file* that passes in any default option values used with the `dsconfig` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `bindDN`
- `bindPasswordFile`
- `certNickname`
- `hostname`
- `keyStorePasswordFile`
- `keyStorePath`
- `port`
- `saslOption`

  SASL is not supported for a proxy server instance.
- `trustAll`
- `trustStorePasswordFile`
- `trustStorePath`
- `useSSL`
- `useStartTLS`

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
dsconfig.trustAll=Yes
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dsconfig`

- Windows: *INSTANCE_DIR*\OUD\bat\dsconfig.bat

Related Commands

- gicadm
- dsreplication

A.1.2.5 dsjavaproperties

The `dsjavaproperties` command specifies the JVM version and Java arguments that are used by each server command.

Synopsis

`dsjavaproperties` [*options*]

Description

The `dsjavaproperties` command can be used to specify the JVM version and Java arguments that are used by each server command. The JVM and Java arguments for each command are specified in a properties file, located at *INSTANCE_DIR*`/OUD/config/java.properties`. The properties file is not used unless you run the `dsjavaproperties` command. If you edit the properties file, you must run `dsjavaproperties` again for the new settings to be taken into account.

`dsjavaproperties` can be used to specify (among other arguments) whether a command runs using the JVM in `-server` mode or `-client` mode. By default, all client applications run in `-client` mode, and all of the server utilities run in `-server` mode. Generally, `-server` mode provides higher throughput than `-client` mode, at the expense of slightly longer startup times.

For certain commands (`import-ldif`, `export-ldif`, `backup`, and `restore`) you can also specify different Java arguments (and a different JVM) depending on whether the command is run in online or offline mode.

If the value of the `overwrite-env-java-home` property is set to `false` in the `java.properties` file, the *OPENDS_JAVA_HOME* environment variable takes precedence over the arguments specified in the properties file. Similarly, if the value of the `overwrite-env-java-args` property is set to `false` in the `java.properties` file, the *OPENDS_JAVA_ARGS* environment variable takes precedence over the arguments specified in the properties file.

Options

The `dsjavaproperties` command accepts an option in either its short form (for example, `-Q`) or their long form equivalent (for example, `--quiet`).

`-Q, --quiet`
Run in quiet mode. Quiet mode does not output progress information to standard output.

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the server and exit rather than attempting to run this command.

Example

The following example shows how to use the `export-ldif` command.

Modifying a Script

This example shows how to change the `export-ldif` script to use a maximum JVM heap size of 256 Mbytes when the command is run with the server online.

1. Edit the *INSTANCE_DIR*`/OUD/config/java.properties` file and set the `export-ldif.online` arguments as follows:

   ```
   export-ldif.online.java-args=-client -Xms8m -Xmx256m
   ```

2. Run the `dsjavaproperties` command for the change to take effect.

   ```
   $ dsjavaproperties
    The script files were successfully updated.  The Oracle Unified Directory
    command-line utilities will use the java properties specified in the
    properties file INSTANCE_DIR/OUD/config/java.properties
   ```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dsjavaproperties`
- Windows: *INSTANCE_DIR*`\OUD\bat\dsjavaproperties.bat`

A.1.2.6 dsreplication

The `dsreplication` command configures replication between directory servers so that the data of the servers is synchronized.

Synopsis

`dsreplication` [*subcommands*] [*options*]

Description

The `dsreplication` command can be used to configure replication between directory servers so that the data of the servers is synchronized. First enable replication by using the `enable` subcommand and then initialize the contents of one directory server with the contents of another server by using the `initialize` subcommand.

The `dsreplication` command contacts the server over SSL using the administration connector (see Managing Administration Traffic to the Server).

Like the `dsconfig` command, `dsreplication` can be run in interactive mode, which walks you through the replication setup process. To run `dsreplication` in interactive mode, type the command name with no parameters, as shown in the following example:

```
$ dsreplication
What do you want to do?

1)  Enable Replication
2)  Disable Replication
3)  Initialize Replication on one Server
```

```
4)  Initialize All Servers
5)  Pre External Initialization
6)  Post External Initialization
7)  Display Replication Status
8)  Purge Historical
9)  Set the trust flag of the Directory Server
10) Enable External Changelog
11) Disable External Changelog
12) Verify Server Configuration
13) List the Certificates Used for Replication
14) Regenerate the Certificate Used for Replication
15) Set the Certificate Used for Replication

c)  cancel

Enter choice: 1
...
```

To display the equivalent non-interactive command, use the `--displayCommand` or `--commandFilePath` option.

Server Subcommands

The following subcommands are used with the `dsreplication` command.

`disable`
Disable replication on the specified directory server for the specified base DN. This subcommand removes references to the specified server in the configuration of the servers with which this server is replicating data. Suboptions are as follows:
`-D, --bindDN` *bindDN*. The DN used to bind to the server on which replication will be disabled. This option must be used if no global administrator has been defined on the server or if you do not want to remove references in the other replicated servers. The password provided for the global administrator is used when this option is specified.
`-a, --disableAll`. Disable the replication configuration on the specified server. The contents of the server are no longer replicated and the replication server (change log and replication port) is disabled, if it is configured.
`--disableReplicationServer`. Disable the replication server. The replication port and change log are disabled on the specified server.
`-h, --hostname` *host*. Directory server host name or IP address.
`-p, --port` *port*. Directory server administration port number.

`disable-changelog`
Disables the external change log for a set of base DNs. If there is no data to replicate, then all the associated replication configuration is removed. For more information about external change log, see Using the External Change Log. Suboptions are as follows:
`-h, --hostname` *host*
Directory server host name or IP address.
`-p, --port` *port*
The Directory Server administration port number.
`-D, --bindDN` *bindDN*
The DN to bind with the server where you want to configure the external change log. The default value is `cn=Directory Manager`.

`enable-changelog`
Creates an external change log for a set of base DNs. The external change log feature allows you to retrieve the modifications performed under a specific base DN. For more information about external change log, see Using the External Change Log. Suboptions are as follows:

`-h, --hostname` *host*
Directory server host name or IP address.
`-p, --port` *port*
The Directory Server administration port number.
`-D, --bindDN` *bindDN*
The DN to bind with the server where you want to configure the external change log. The default value is `cn=Directory Manager`.
`-r, --replicationPort` *port*
The port required to configure the change log. You must specify this option only if the changelog (or replication) is not previously configured in the server. The default value is `8989`.

`enable`
Update the configuration of the directory servers to replicate data under the specified base DN. If one of the specified servers is already replicating the data under the base DN to other servers, executing this subcommand updates the configuration of all the servers. It is therefore sufficient to execute the subcommand once for each server that is added to the replication topology. Suboptions are as follows:
`--bindDN2` *bindDN*. The DN used to bind to the second server whose contents will be replicated. If no bind DN is specified, the global administrator is used to bind.
`--bindPasswordFile1` *filename*. The file containing the password used to bind to the first server whose contents will be replicated. If no bind DN was specified for the first server, the password of the global administrator is used to bind.
`-D, --bindDN1` *bindDN*. The DN used to bind to the first server whose contents will be replicated. If no bind DN is specified, the global administrator is used to bind.
`-F, --bindPasswordFile2` *filename*. The file containing the password used to bind to the second server whose contents will be replicated. If no bind DN was specified for the second server, the password of the global administrator is used to bind.
`-h, --host1` *host*. Host name or IP address of the first server whose contents will be replicated.
`--noReplicationServer1`. Do not configure a replication port or change log on the first server. The first server will contain replicated data but will not contain a change log of modifications made to the replicated data. Each replicated topology must contain at least two servers with a change log to avoid a single point of failure.
`--noReplicationServer2`. Do not configure a replication port or change log on the second server. The second server will contain replicated data but will not contain a change log of modifications made to the replicated data. Each replicated topology must contain at least two servers with a change log to avoid a single point of failure.
`--noSchemaReplication`. Do not replicate the schema between the servers. (Schema replication is enabled by default.) Use this option if you do not want the schema to be synchronized between servers.
`--onlyReplicationServer1`. Configure only a change log and replication port on the first server. The first server will not contain replicated data, but will contain a change log of the modifications made to the replicated data on other servers.
`--onlyReplicationServer2`. Configure only a change log and replication port on the second server. The second server will not contain replicated data, but will contain a change log of the modifications made to the replicated data on other servers.
`-O, --host2` *host*. Hostname or IP address of the second server whose contents will be replicated.
`-p, --port1` *port*. Directory server administration port number of the first server whose contents will be replicated.
`--port2` *port*. Directory server administration port number of the second server whose contents will be replicated.

`-r, --replicationPort1` *port*. The port that will be used by the replication mechanism in the first directory server to communicate with other servers. Only specify this option if replication was not previously configured on the first directory server.
`-R, --replicationPort2` *port*. The port that will be used by the replication mechanism in the second directory server to communicate with other servers. Only specify this option if replication was not previously configured in the second server.
`-S, --skipPortCheck`. Skip the check to determine whether the specified replication ports are usable. If this argument is not specified, the server checks that the port is available only if you are configuring the local host.
`--secureReplication1`. Specifies whether communication through the replication port of the first server is encrypted. This option is only taken into account the first time replication is configured on the first server.
`--secureReplication2`. Specifies whether communication through the replication port of the second server is encrypted. This option is only taken into account the first time replication is configured on the second server.
`--useSecondServerAsSchemaSource`. Use the second server to initialize the schema of the first server. If neither this option nor the `--noSchemaReplication` option is specified, the schema of the first server is used to initialize the schema of the second server.

`initialize`
Initialize the contents of the data under the specified base DN on the destination directory server with the contents on the source server. This operation is required after enabling replication. Suboptions are as follows:
`-h, --hostSource` *host*. Directory server host name or IP address of the source server whose contents will be used to initialize the destination server.
`-O, --hostDestination` *host*. Directory server hostname or IP address of the destination server whose contents will be initialized.
`-p, --portSource` *port*. Directory server administration port number of the source server whose contents will be used to initialize the destination server.
`--portDestination` *port*. Directory server administration port number of the destination server whose contents will be initialized.

`initialize-all`
Initialize the data under the specified base DN, on all the directory servers in the topology, with the data on the specified server. This operation is required after enabling replication for replication to work. Alternatively, you can use the `initialize` subcommand on each individual server in the topology. Suboptions are as follows:
`-h, --hostname` *host*. Directory server host name or IP address of the source server.
`-p, --port` *port*. Directory server administration port number of the source server.

`list-certs`
List the certificates used by the servers for replication. Suboptions are as follows:
`-h, --hostname` *host*
Directory server host name or IP address.
`-p, --port` *port*
Directory server administration port number. Default value: 4444

`post-external-initialization`
Enable replication to work after the entire topology has been reinitialized by using `import-ldif` or binary copy. This subcommand must be called after you initialize the contents of all directory servers in a topology by using `import-ldif` or binary copy. If you do not run this subcommand, replication will no longer work after the initialization. Suboptions are as follows:
`-h, --hostname` *host*. Directory server host name or IP address.
`-p, --port` *port*. Directory server administration port number.

pre-external-initialization
Prepare a replication topology for initialization by using `import-ldif` or binary copy. This subcommand must be called before you initialize the contents of all directory servers in a topology by using `import-ldif` or binary copy. If you do not run this subcommand, replication will no longer work after the initialization. After running this subcommand, initialize the contents of all the servers in the topology, then run the subcommand `post-external-initialization`. Suboptions are as follows:
`-h, --hostname` *host*. Directory server host name or IP address.
`-l, --local-only`. Use this option when the contents of only the specified directory server will be initialized with an external method.
`-p, --port` *port*. Directory server administration port number.

purge-historical
Launches a purge processing of the historical information stored in the user entries by replication. Since this processing may take a while, you must specify the maximum duration for this processing. Suboptions are as follows:
`-h, --hostname` *host*. Directory server host name or IP address.
`-p, --port` *port*. Directory server administration port number.
`--maximumDuration` *maximum duration*. Specifies the maximum duration the purge processing must last expressed in seconds. The default value is 3600.
`-t, --start` *startTime*. Specifies the date and time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. Use `0` to schedule the task for immediate execution. When this option is specified the operation is scheduled to start at the specified time after which the utility exits immediately.
`--recurringTask` *schedulePattern*. Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.
`--completionNotify` *emailAddress*. Indicates the e-mail address of the recipient to be notified when the task completes. You can specify this option more than once.
`--errorNotify` *emailAddress*. Indicates the e-mail address of the recipient to be notified if an error occurs when this task executes. You can specify this option more than once.
`--dependency` *taskID*. Indicates the ID of a task upon which this task depends. A task will not start execution until all its dependent tasks have completed execution.
`--failedDependencyAction` *action*. Indicates the action that should take place if one if its dependent tasks fail. It must have one of the following values: `PROCESS`,`CANCEL`, or `DISABLE`. The default value is CANCEL.

regenerate-cert
Regenerates the certificate used by the specified server (or all servers) for replication. Suboptions are as follows:
`-a, --all`
Regenerates the certificate of all servers configured for replication (and not only of the server provided to connect).
`-h, --hostname` *host*
Directory server host name or IP address.
`-p, --port` *port*
Directory server administration port number. Default value: 4444

set-cert
Configures the server to use a certificate in a keystore for replication. Suboptions are as follows:
`--replCertNickName` *nickname*
Specifies the nickname of the certificate that you want to be used by the server for replication.
`--replKeyStoreType` *type*

Specifies the type of the keystore. The value can be any type of keystore, including JKS, JCEKS, PKCS12, and PKCS11. The Java Virtual Machine used by the server must support this keystore type (by default, most JVMs support the keystore types JKS, JCEKS, and PKCS12). The default value is JKS.

`--replKeyStorePath` *path*

Specifies the path of the keystore containing the certificate to be used by the server for replication. This value is not required if the certificate is stored on a hardware device such as a Java card. The server must have read access rights to this path. You can specify a path relative to the location of the server (for example, `config/my-keystore`).

`--replKeyStorePasswordFile` *path*

Specifies the path to the file containing the password (PIN) needed to access the keystore. The password must be stored in clear text in the file, and the server must have read access rights to the file. If you want the server to encrypt the contents of the file, the server must also have write access rights. You can specify a path relative to the location of the server (for example, `config/my-keystore.pin`).

`--replKeyPasswordFile` *keyPasswordFile*

Specifies the file containing the password (PIN) needed to access the private key of the certificate in the keystore. This option is required if the private key cannot be read using the keystore password specified with the `--replKeyStorePasswordFile` option. The password must be stored in clear text in the file, and the server must have read access rights to the file. You can specify a path relative to the location of the server (for example, `config/my-key.pin`).

`--skipLocalChecks`

Specifies that the default checks to validate the provided data should be skipped when you run the command from the same machine as the server.

`--encryptKeyStorePasswordFile`

Specifies that the server should overwrite the contents of the password (PIN) file you provide with the password encrypted. The password file must contain the password in clear text, and the server must have write access rights on the file specified using the `--replKeyStorePasswordFile` option.

`-h, --hostname` *host*

Directory server host name or IP address.

`-p, --port` *port*

Directory server administration port number. Default value: 4444

`set-trust`

Set the trust flag of a directory server. Any change that is sent by an untrusted directory server will be discarded by the rest of the topology. Only trusted directory servers are allowed to send changes to be replayed by other directory servers. Suboptions are as follows:

`-h, --trustedHost` *host*. Specifies the fully qualified host name or IP address of the directory server that will perform the change.

`-p, --trustedPort` *port*. Specifies the administration port number of the directory server that will perform the change.

`-M, --modifiedHost` *host*. Specifies the fully qualified host name or IP address of the directory server whose trust flag is modified.

`-c, --modifiedPort` *port*. Specifies the administration port number of the directory server whose trust flag is modified.

`-t, --trustValue` *trusted*|*untrusted*. Specifies the new value of the trust flag for the directory server to be modified. The value can be `trusted` or `untrusted`. The default value is `trusted`.

`status`

List the replication configuration for the specified base DNs of all directory servers defined in the registration information. If no base DNs are specified, the information for all base DNs is displayed. Suboptions are as follows:

`-h, --hostname` *host*

Directory server host name or IP address.
`-p, --port` *port*
Directory server administration port number. Default value: `4444`
`-D, --bindDN` *bindDN*
DN to use to bind to the server if no global administrator has been defined. This can be used to view the external changelog status (which does not require replication between servers to be configured). Default value: `cn=Directory Manager`
`-e, --expanded`
Use expanded view of the replication status showing the replication domains (replicated data) and replication servers (change log and replication port) as separate entities.
`-s, --script-friendly`
Use the script-friendly mode.
`-d, --dataToDisplay {dataToDisplay}`
Specify the replication data information that you want to be displayed. For example, if you enter the following:
`--dataToDisplay entry-number --dataToDisplay missing-changes`
the number of entries and the missing changes is displayed. For more information about the different values allowed, run the following:

```
dsreplication status --listDataToDisplay
```

`--listDataToDisplay`
List the different values that can be used for the argument `--dataToDisplay`. If you want to display the full list of values, also specify `--advanced` argument.

`verify`
Verifies the replication configuration of the different replicated servers.
Oracle recommends that you run the `verify` subcommand in interactive mode (without the `--no-prompt` option). Then, if any inconsistencies are found in the replication configuration, they will be displayed and you can fix them interactively.
For example, you can use the `verify` subcommand:

- To remove references to servers that are no longer reachable (for example, because they crashed and are not recoverable or they were not properly uninstalled).
- To fix configuration problems related to the certificates used by the replication system.
- To update the host names used by the replication configuration.

Suboptions are as follows:
`-r, --replicationServer` *hostName*:*replicationPort*
Specifies the host names and replication ports used in the configuration to reference the replication servers. These values are applied to all servers that are replicated. For example:
`replicationhost1.example.com:8989`
`--noReplicationDomainUpdate`
If specified when you provide a list of replication servers using the `--replicationServer` argument, the changes are not applied to the replication domains.
`-a, --serverToAdd` *hostName*:*administrationPort*
Specifies servers that have been removed from the registration information by mistake and must be added again (for example, servers removed by using the `--serverToRemove` option by mistake). If there are several network interfaces defined in the server, use commas to separate them. For example:
`host1-interface1.domain1.com,host2-interface2.domain2.com:4444`
`-s, --serverToRemove` *hostName*:*administrationPort*
Specifies servers that are not reachable (for example, servers that have been uninstalled), but they are still referenced by the other servers and are mentioned when running `dsreplication`

`status`. The references to these servers will be removed. For example: `replicationhost3.example.com:4444`

`--updateAddress` *`oldHostName/newHostName1,newHostName2`*

Specifies new addresses that the registration information should use for a given server. Use this argument when a network interface has changed or when the complete list of addresses was not provided when replication was configured, which can result in duplicate lines when running `dsreplication status`.

You must specify the addresses currently used for the server and the new servers that you want to use. For example: `oldhost/newhost1,newhost2`

`--fixCertificates`

Fixes any problems found with the certificates that are used by the replication to communicate between servers.

`-h, --hostname` *`host`*

Directory server host name or IP address.

`-p, --port` *`port`*

Directory server administration port number. Default value: 4444

Options

The `dsreplication` command accepts an option in either its short form (for example, `-H`) or its long form equivalent (for example, `--help`).

`-b, --baseDN` ***`baseDN`***

Specify the base DN of the data to be replicated or initialized, or for which replication should be disabled. Multiple base DNs can be specified by using this option multiple times. Use `virtual-acis` if you want to replicate the virtual ACIs.

Configuration Options

`--advanced`

Use this option to access advanced settings when running this command in interactive mode.

LDAP Connection Options

`-I, --adminUID` ***`adminUID`***

Specify the User ID of the global administrator to bind to the server. If no global administrator was defined previously for any of the servers, this option creates a global administrator by using the data provided.

`-j, --adminPasswordFile` ***`bindPasswordFile`***

Use the global administrator password in the specified file when authenticating to the directory server.

`-o, --saslOption` ***`name=value`***

Use the specified options for SASL authentication.
SASL is not supported for a proxy server instance.

`-X, --trustAll`

Trust any certificate that the server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

`-P, --trustStorePath` ***`trustStorePath`***

Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-U, --TrustStorePasswordFile` ***path***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-K, --keyStorePath` ***keyStorePath***
Use the client keystore certificate in the specified path.

`-u, --keyStorePasswordFile` ***keyStorePasswordFile***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-N, --certNickname` ***nickname***
Use the specified certificate for authentication.

`--connectTimeout` ***timeout***
Specifies the maximum length of time (in milliseconds) that can be taken to establish a connection. Use `0`to specify no time out. The default value is 30000.

Command Input/Output Options

`--commandFilePath` ***path***
Specify the full path to the file in which the equivalent non-interactive commands are written when the command is run in interactive mode.

`--displayCommand`
Display the equivalent non-interactive command in the standard output when the command is run in interactive mode.

`-n, --no-prompt`
Run in non-interactive mode. If some data in the command is missing, the user will not be prompted and the command will fail.

`--noPropertiesFile`
Indicate that the command will not use a properties file to get the default command-line options.

`--propertiesFilePath` ***propertiesFilePath***
Specify the path to the properties file that contains the default command-line options.

`-Q, --quiet`
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples assume that two directory servers are installed: `host1` and `host2`. Both servers are configured with the default administration port (4444). The base DN

`dc=example,dc=com` is populated with data on `host1`. The base DN exists on `host2`, but is empty. The examples configure replication between the two servers and initialize `host2` with data.

> **Note:**
>
> The easiest way to use `dsreplication` is in interactive mode, in which case you are prompted for all of the relevant arguments. Using the interactive mode and `--displayCommand` option, you can get the equivalent non-interactive command to do scripting).
>
> To illustrate which arguments are configured, these examples do not use the interactive mode.

Enabling Directory Server Replication

The following command enables replication for the base DN `dc=example,dc=com` on `host1` and `host2`. The command runs in non-interactive mode (`-n`) and specifies that all server certificates should be accepted (`-X`).

```
$ dsreplication enable \
  --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPasswordFile1 /tmp/pwd-file --replicationPort1 8989 \
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPasswordFile2 /tmp/pwd-file --replicationPort2 8989 \
  --adminUID admin --adminPasswordFile /tmp/pwd-file --baseDN "dc=example,dc=com" -X -n
```

Initializing Directory Server Replication

To initialize one replica from another, use the `initialize` subcommand. The following command initializes the base DN `dc=example,dc=com` on `host2` with the data contained on `host1`. The command runs in non-interactive mode (`-n`) and specifies that all server certificates should be accepted (`-X`).

```
$ dsreplication initialize --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPasswordFile /tmp/pwd-file \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 -X -n
```

To initialize an entire topology, use the `initialize-all` subcommand. This subcommand takes the details of the source directory server as options and initializes all other replicas for which replication has been enabled.

Obtaining the Directory Server Replication Status

The following command obtains the replication status of the directory servers in the topology.

```
bin/dsreplication status -X --adminPasswordFile /tmp/password.txt -n
Establishing connections .......... Done.

dc=example,dc=com - Replication Enabled
========================================
Server             : Entries : M.C. [1] : A.O.M.C. [2] : Port [3] : Status [4]
-------------------:---------:----------:--------------:----------:-------------
host1:4444         : 200000  : 0        : N/A          : 8989     : Normal
host2:5444         : 200000  : 0        : N/A          : 9989     : Normal

[1] The number of changes that are still missing on this element (and that have been
```

```
applied to at least one other server).
[2] Age of oldest missing change: the age (in seconds) of the oldest change that has not
yet arrived on this element.
[3] The replication port used to communicate between the servers whose contents are
being replicated.
[4] The status of the replication on this element.



To have the same output as in previous versions, the user must use the --dataToDisplay
argument with the 'compat-view' value:

bin/dsreplication status -X --adminPasswordFile /tmp/password.txt -n --dataToDisplay
compat-view
Establishing connections ......... Done.

dc=example,dc=com - Replication Enabled
=======================================
Server     : Entries : M.C. [1] : A.O.M.C. [2] : Port [3] : SSL [4]  : Trust [5] : U.C.
[6] : Status [7]   : ChangeLog [8] : Group ID [9] : Connected To [10]
-----------:---------:----------:--------------:----------:----------:-----------:-------
---:--------------:--------------:--------------:---------------------------
host1:4444 : 200000  : 0        : N/A          : 8989     : Disabled : Trusted   :
--       : Normal       : Enabled       : 1            : host1:8989 (GID=1)
host2:5444 : 200000  : 0        : N/A          : 9989     : Disabled : Trusted   :
--       : Normal       : Enabled       : 1            : host2:9989 (GID=1)

[1] The number of changes that are still missing on this element (and that have been
applied to at least one other server).
[2] Age of oldest missing change: the age (in seconds) of the oldest change that has not
yet arrived on this element.
[3] The replication port used to communicate between the servers whose contents are
being replicated.
[4] Whether the replication communication initiated by this element is encrypted or not.
[5] Whether the directory server is trusted or not. Updates coming from an untrusted
server are discarded and not propagated.
[6] The number of untrusted changes. These are changes generated on this server while it
is untrusted.
    Those changes are not propagated to the rest of the topology but are effective on
the untrusted server.
[7] The status of the replication on this element.
[8] Whether the external change log is enabled or not for the base DN on this server.
[9] The ID of the replication group to which the server belongs.
[10] The replication server this element is connected to with its group ID between
brackets.
```

Disabling Directory Server Replication

The following command disables replication for the base DN `dc=example,dc=com` on `host2`. Disabling replication on one directory server removes all references to that server from the other directory servers in the replication topology.

```
$ dsreplication disable --baseDN "dc=example,dc=com" \
--hostname host2 --port 4444 --adminUID admin --adminPasswordFile /tmp/pwd-file \
-X -n
Establishing connections ..... Done.
Disabling replication on base DN cn=admin data of server host2:4444 ..... Done.
Disabling replication on base DN dc=example,dc=com of server host2:4444 ..... Done.
Disabling replication on base DN cn=schema of server host2:4444 ..... Done.
Removing references on base DN cn=admin data of server host1:4444 ..... Done.
Removing references on base DN dc=example,dc=com of server host1:4444 ..... Done.
```

```
Removing references on base DN cn=schema of server host1:4444 ..... Done.
Disabling replication port 8990 of server host2:4444 ..... Done.
```

Configuring the External Change Log on a Non-replicated Server

The following example illustrates the replication status before enabling the change log:

```
bin/dsreplication status -X --adminPasswordFile /tmp/password.txt -n --bindDN
"cn=directory manager" --port 4444
Establishing connections ..... Done.

dc=example,dc=com - Replication Disabled
========================================
Server        : Entries   : ChangeLog [1]
--------------:-----------:--------------
host1:4444    : 200000    : Disabled
```

The following command enables the external change log on a non-replicated server.

```
bin/dsreplication enable-changelog -X --adminPasswordFile /tmp/password.txt -n --bindDN
"cn=directory manager" -b dc=example,dc=com
Establishing connections ..... Done.

Configuring Replication port on server host1:4444 ..... Done.
Enabling Changelog on base DN dc=example,dc=com ..... Done
```

The following example illustrates the replication status after changelog has been enabled:

```
bin/dsreplication status -X --adminPasswordFile /tmp/password.txt -n --bindDN
"cn=directory manager" --port 4444
Establishing connections ..... Done.

dc=example,dc=com - Replication Disabled
========================================
Server        : Entries   : ChangeLog [1]
--------------:-----------:--------------
host1:4444    : 200000    : Enabled

[1] Whether the external change log is enabled for the base DN on this server or not.
```

Exit Codes

0
Successful.

1
Unable to initialize arguments.

2
Cannot parse arguments because the provided arguments are not valid or there was an error checking the user data.

3
The user canceled the operation in interactive mode.

4
Conflicting arguments.

5
The specified base DNs cannot be used to enable replication.

`6`
The specified base DNs cannot be used to disable replication.

`7`
The specified base DNs cannot be used to initialize the contents of the replicas.

`8`
Error connecting with the credentials provided.

`9`
Could not find the replication ID of the domain to be used to initialize the replica.

`10`
The maximum number of attempts to start the initialization has been exceeded. A systematic "peer not found error" was received.

`11`
Error enabling replication on base DN.

`12`
Error initializing base DN.

`13`
Error reading configuration.

`14`
Error updating ADS.

`15`
Error reading ADS.

`16`
Error reading Topology Cache.

`17`
Error configuring the replication server.

`18`
Unsupported ADS scenario.

`19`
Error disabling replication on base DN.

`20`
Error removing replication port reference on base DN.

`21`
Error initializing Administration Framework.

`22`
Error seeding trust store.

`23`
Error launching pre-external initialization.

`24`
Error launching post-external initialization.

25
Error disabling replication server.

26
Error executing purge historical.

27
The specified base DN cannot be purged.

28
Error launching purge historical.

29
Error loading configuration class in local purge historical.

30
Error starting server in local purge historical.

31
Timeout error in local purge historical.

32
Generic error executing local purge historical.

33
The trusted host was not found in the ADS.

34
The modified host was not found in the ADS.

35
The changelog cannot be enabled on this base DN.

36
The changelog cannot be disabled on this base DN.

37
An error occurred configuring the changelog.

38
The specified host was not found in the configuration.

39
No base DN available to enable replication. This occurs when you request to enable replication between two servers, and the two servers do not have common base DNs to configure replication. It may be that they are already replicated, or that simply they are not defined).

40
No base DNs replicated. The server does not contain any replicated base DN. Thus the operations requiring replicated base DNs, for example `initialize`, cannot be applied to the server.

41
A source for the initialization has been specified, but no destination server could be found. None of the other servers are replicating a base DN with the server chosen as source).

42
There are replication servers or replication domains with the same replication ID. This occurs, for instance, when you try to merge two replication topologies.

43
An unidentified error.

44
Error configuring crypto manager (updating the secure connection configuration for replication).

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `dsreplication` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `adminUID`
- `baseDN`
- `certNickname`
- `keyStorePasswordFile`
- `keyStorePath`
- `saslOption`

  SASL is not supported for a proxy server instance.
- `trustAll`
- `trustStorePasswordFile`
- `trustStorePath`

```
toolname.propertyname=propertyvalue
```

Entries in the properties file have the following format:

For example:

```
dsreplication.baseDN=dc=example,dc=com
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dsreplication`
- Windows: *INSTANCE_DIR*`\OUD\bat\dsreplication.bat`

Related Commands

- dsconfig

A.1.2.7 dstune

The `dstune` utility allows you to tune the Oracle Unified Directory server.

Synopsis

`dstune` [*subcommand*] [*options*]

Description

The `dstune` utility can be used to help you tuning the Oracle Unified Directory server based on criteria such as a memory limitation or the data that it will contain. To tune the server, you must use a Java Virtual Machine that uses Java HotSpot(TM), such as the Oracle Java Standard Edition.

Subcommands

The `dstune` utility provides the subcommands described in this section.

> **Note:**
>
> The `dstune automatic` subcommand is no longer available (`automatic` subcommand usage is still available for backward compatibility).
>
> To specify automatic tuning similar to previous versions of Oracle Unified Directory, use the `dstune set-runtime-options` subcommand with the `--value autotune` suboption.

`data-based`
Tunes the server using information describing the data that the directory server will contain. Use `--ldifFile` to do the tuning based in the contents of an LDIF file. Use `--entryNumber` and `--entrySize` to do the tuning based on the number of entries and size.
If you do not specify any of these options, the server will be tuned using the data that the server currently contains.
Suboptions are as follows:
`-l, --ldifFile` *`path`*
Specifies the path of the LDIF file whose contents will be used to tune the server.
`--entryNumber` *`entrynumber`*
Specifies the number of entries that the Oracle Unified Directory server will contain. This value and the value provided for `--entrySize` will be used to tune the server. The default value is `100000`.
`--entrySize` *`entrysize`*
Specifies the average size in kilobytes of the entries that the Oracle Unified Directory server will contain. This value and the value provided for `--entryNumber` will be used to tune the server. The default value is `4`.

`list`
Lists the tuning settings for the server.

`mem-based`
Specifies the memory to be used for the tools you want to tune.
Suboptions are as follows:
`--memory { heap-size | system memory | system memory percentage }`
Specifies the memory to be used for the tools to be tuned.

To tune the tools based on the heap size of their respective Java process, provide a value for the memory (for example, `768m`). For gigabytes, use `g` after the value (for example, `2.5g`). If you do not specify a unit after the value, megabytes will be used.
If you are tuning the server, you can specify the system memory (`systemMemory`) as either an amount or percentage:

- Amount: For example, to use 2 GB, specify `systemMemory:2g`. The `dstune` utility then splits the value you provide into two parts: the heap size that the Java Virtual Machine of the server will use and an estimation of the required file-system cache.
- Percentage: For example, to dedicate 50 percent of system memory to the server, specify `systemMemory:50.0%`. To fully dedicate a machine to the server, specify `systemMemory:100%`.

The default value for the memory will be calculated based on your specific configuration and the free memory available on the system where `dstune` is running.
`--targetTool { server | import-ldif | export-ldif | rebuild-index | verify-index | all }`
Specifies the tools that should be tuned. The default is `server`.

`set-runtime-options`
Tunes the tools using the provided runtime settings. Use `--value` to provide the value of the runtime settings and `--targetTool` to provide the tools the value should be applied to. Suboptions are as follows:
`--value { autotune | jvm-default | JVM arguments }`
Specifies the tuning parameters for the tools. The tools can be automatically tuned each time they are launched based on the available memory in the machine (provide the value `autotune`), the tools can use the default Java Virtual Machine on your system to choose the runtime settings (provide the value `jvm-default`), or you can provide directly the Java arguments that the tools must use (for instance '`-server -Xmx1024m`'). The default is `autotune`.
`--targetTool { server | import-ldif | export-ldif | rebuild-index | verify-index | all }`
Specifies the tools that should be tuned. The default is `server`.

Utility Input/Output Options

`-Q, --quiet`
Perform a quiet operation (no progress information is written to the standard output).

`-n, --no-prompt`
Run utility in non-interactive mode. If some data in the command is missing, the user will not be prompted and the command will fail.

`-v, --verbose`
Use verbose mode.

`--displayCommand`
Display the equivalent non-interactive option in the standard output when this command is run in interactive mode.

`--commandFilePath` ***path***
Specify the full path to the file, where the equivalent non-interactive commands will be written when this command is run in interactive mode.

`--propertiesFilePath` ***propertiesFilePath***
Specify the path to the properties file that contains the default command-line options.

--noPropertiesFile
Indicate that the command will not use a properties file to get the default command-line options.

General Options

-V, --version
Displays the version information for the directory server.

-?, -H, --help
Displays command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

Examples

The examples in this section show how to use the `dstune` utility to tune the Oracle Unified Directory server and tools (`import-ldif`, `export-ldif`, `verify-index`, and `rebuild-index`).

> **Note:**
>
> Beginning with Oracle Unified Directory 11g Release 2 (11.1.2.3), the `dstune automatic` subcommand is no longer available (`automatic` subcommand usage is still available for backward compatibility).
>
> To specify automatic tuning similar to previous versions of Oracle Unified Directory, use the `dstune set-runtime-options` subcommand with the `--value autotune` suboption.
>
> See dstune

Memory-Based Tuning

The following subcommand tunes the server and all tools specifying 2 GB for the heap size.

```
$ dstune mem-based --memory 2g --targetTool all

Calculating Tuning Settings ..... Done.
Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

Data-Based Tuning

The following subcommand tunes the server based on the assumption that the server contains `10000000` entries with an average size of `20` KB each.

```
$ dstune data-based --entryNumber 10000000 --entrySize 20

Calculating Tuning Settings ..... Done.
Updating the tuning properties ..... Done.
Updating scripts ..... Done.
```

Runtime Tuning

The following subcommand tunes only the server and `import-ldif` tool to use automatic tuning. Each time you run the server and the `import-ldif` tool, tuning is done based on the system resources.

```
$ dstune set-runtime-options --targetTool server --targetTool import-ldif \
--value autotune

Updating the tuning properties ..... Done.

The server will be automatically tuned the next time it will be restarted.
```

Displaying the Current Tuning Mode

The following subcommand displays the current tuning settings for an Oracle Unified Directory server instance.

```
$ dstune list

Tool          : Tuning Value
--------------:-----------------
server        : -server
import-ldif   : -Xmx2048m -server
export-ldif   : Automatic Tuning
rebuild-index : Automatic Tuning
verify-index  : -Xmx2048m -server
```

Exit Codes

0
The operation was completed successfully, this includes the cases where no operation is performed with no errors (for instance, the usage was displayed).

1
Unable to initialize the arguments.

2
The data provided by the user was not correct (for instance, invalid values or conflicting attributes).

3
The user canceled the operation during interaction.

4
Error writing the java.properties file.

5
Error executing the `dsjavaproperties` command-line to update the tuning settings.

6
An error occurred retrieving the JVM tuning settings. This occurs when the algorithms used by `dstune` of are not able to find some valid settings.

7
An error occurred reading the java.properties file while displaying the current tuning settings.

8
An unidentified error.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dstune`
- Windows: *INSTANCE_DIR*`\OUD\bat\dstune.bat`

Related Commands

- dsjavaproperties

A.1.2.8 gicadm

The `gicadm` command manages global indexes and global index catalogs.

This command is supported only for the proxy.

Synopsis

`gicadm` [*subcommand*] [*options*]

Description

The `gicadm` command enables you to create and delete a global index catalog, as well as add, modify, and delete global indexes in a global index catalog, and manage replication of global index catalogs. It also allows you to associate a global index to a distribution.

The `gicadm` command accesses the server over SSL through the administration connector.

Options

The `gicadm` command accepts the following options.

`add-index`
Adds a new global index to a global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--attributeName attribute-name`. The identifier for the global index attribute. This identifier should be unique in the context of the global index catalog and it is used to identify the global index.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it.

`associate`
Associates a global index catalog to a distribution workflow element. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`-d, --distributionWorkflowElement` *distribution-workflow-element*. The name of the distribution workflow element object using this global index catalog, from which the global index catalog is to be disassociated.

`create-catalog`
Creates a new global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.

`delete-catalog`
Deletes a global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.

disable-replication
Disables replication on the specified server for the specified global index catalog and removes any references to this server from the other servers in the replication topology. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--adminUID` *adminUID*. User ID of the global administrator used to bind to the server. For the `enable-replication` subcommand if no global administrator was defined previously the global administrator will be created using the provided data.

disassociate
Disassociates a global index catalog from a distribution workflow element. Suboptions are as follows:
`-d, --distributionWorkflowElement` *distribution-workflow-element*. The name of the distribution workflow element object using this global index catalog, from which the global index catalog is to be disassociated.

enable-replication
Updates the server configuration to replicate the global index catalog and all its global indexes. If one of the specified servers already replicates the global index catalog for a given global index, executing this subcommand will update the configuration of all servers in the topology. Therefore, it is sufficient to execute this command once for each server added to the replication topology. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--adminUID` *adminUID*. User ID of the global administrator used to bind to the server. For the `enable-replication` subcommand, if no global administrator was defined previously, the global administrator will be created using the provided data.
`--adminPasswordFile` *bindPasswordFile*. The file containing the password of the global administrator.
`--localReplicationPort` *port*. Replication port number of the first server whose content will be replicated.
`--localSecureReplication`. Specifies whether the communication through the replication port of the first server is encrypted or not. This option will only be taken into account the first time replication is configured on the first server.
`--remoteAdminPort` *port*. Directory server administration port number of the second server whose contents will be replicated.
`--remoteHost` *host*. Fully qualified directory server host name or IP address of the second server whose contents will be replicated.
`--remoteBindDN` *bindDN*. DN to use to bind to the second server whose content will be replicated. If not specified the global administrator will be used to bind.
`--remoteBindPasswordFile` *bindPasswordFile*. File containing the password to use to bind to the second server whose content will be replicated. If no bind DN was specified for the second server the password of the global administrator will be used to bind.
`--remoteReplicationPort` *port*. Replication port number of the second server whose content will be replicated.
`--remoteSecureReplication`. Specifies whether the communication through the replication port of the second server is encrypted or not. This option will only be taken into account the first time.

export
Exports a global index catalog to file. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.

`--exportDirectory` *directory*. Path to the directory to be used to export the global index catalog. This is a required argument.
`-a, --attributeName attribute-name`. The name of the global index attribute. This option can be used multiple times to specify multiple indexed attributes. If this option is provided, any indexed attribute in the import source that does not match is skipped.

`get-catalog-prop`
Shows global index catalog properties. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--property` *property*. The name of a property to be displayed.
`-E,--record`. Modifies the display output to show one property value per line.

`get-index-prop`
Shows index properties. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`-a, --attributeName` *attribute-name*. The identifier for the global index attribute. This identifier should be unique in the context of the global index catalog and it is used to identify the global index.
`--property` *property*. The name of a property to be displayed. Valid property names are:`all`, `global-index-deleted-entry-retention-timeout`, `db-cleaner-min-utilization`, `db-log-file-max`, `db-checkpointer-bytes-interval`, `db-checkpointer-wakeup-interval`, `db-num-lock-tables`, `db-num-cleaner-threads`, `db-txn-no-sync`, `db-txn-write-no-sync`, `je-property`, `db-directory`, `db-directory-permissions`, `global-index-catalogs-shared-cache`, and `global-index-attribute`.

`import`
Imports content of a file into a specified global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--importDirectory` *directory*. Path to the file to be used to import the global index catalog. This is a required argument.
`--attributeName` *attribute-name*. The identifier for the global index attribute. This identifier should be unique in the context of the global index catalog and it is used to identify the global index.
`--append`. Append to an existing global index rather than overwriting it.

`initialize-replication`
Initializes the replication of a global index catalog. All the replicated global index catalogs (part of the replication topology) can be initialized at once or the local global index catalog is initialized from a given global index catalog (also part of the replication topology). Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--adminUID` *adminUID*. User ID of the global administrator used to bind to the server. For the `initialize-replication` subcommand, if no global administrator was defined previously, the global administrator will be created using the provided data.
`--fromServerPort` *port*. Directory server port number of the source server whose contents will be used to initialize the destination server.
`--fromServerHost` *host*. Directory server hostname or IP address of the source server whose contents will be used to initialize the destination server.
`--all`. Initializes the contents of the global index attribute on all the servers whose contents is being replicated with the contents on the specified server.

list-catalogs
Lists the global index catalogs that have been defined. Suboptions are as follows:
`--property` *property*. The name of a property to be displayed. Valid property names are:`all`, `replication-server`, `server-id`, `window-size`, `heartbeat-interval` and `group-id`.

list-indexes
Lists the global indexes that have been defined in the global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--property` *property*. The name of a property to be displayed. Valid property names are:`all`, `global-index-deleted-entry-retention-timeout`, `db-cleaner-min-utilization`, `db-log-file-max`, `db-checkpointer-bytes-interval`, `db-checkpointer-wakeup-interval`, `db-num-lock-tables`, `db-num-cleaner-threads`, `db-txn-no-sync`, `db-txn-write-no-sync`, `je-property`, `db-directory`, `db-directory-permissions`, `global-index-catalogs-shared-cache`, and `global-index-attribute`.

post-external-initialization
This subcommand must be called after initializing the contents of all the replicated global indexes using the import subcommand of this tool. It will use the generation id of the targeted instance as the valid one. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`-a, --attributeName` *attribute-name*. The identifier for the global index attribute. This option can be used multiple times to specify multiple indexed attributes. If this option is provided, any indexed attribute in the import source that does not match is skipped.

pre-external-initialization
This subcommand can be called before initializing the contents of all the replicated servers using the import subcommand of this tool. It will erase the replication change logs stored in the replication servers. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`-a, --attributeName` *attribute-name*. The identifier for the global index attribute. This option can be used multiple times to specify multiple indexed attributes. If this option is provided, any indexed attribute in the import source that does not match is skipped.

remove-index
Removes a global index from a global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--attributeName` *attribute-name*. The identifier for the global index attribute. This identifier should be unique in the context of the global index catalog and it is used to identify the global index.

set-catalog-prop
Modifies the properties of the global index catalog. Suboptions are as follows:
`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--set` *property*`:`*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it. Valid property names are: `all`, `global-index-deleted-entry-retention-timeout`, `db-cleaner-min-utilization`, `db-log-file-max`, `db-checkpointer-bytes-interval`, `db-checkpointer-wakeup-interval`, `db-num-lock-tables`, `db-num-cleaner-threads`, `db-txn-no-sync`, `db-txn-write-no-sync`, `je-property`, `db-`

directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.
--reset *property*. Resets a property back to its default values, where *property* is the name of the property to be reset. Valid property names are: all, global-index-deleted-entry-retention-timeout, db-cleaner-min-utilization, db-log-file-max, db-checkpointer-bytes-interval, db-checkpointer-wakeup-interval, db-num-lock-tables, db-num-cleaner-threads, db-txn-no-sync, db-txn-write-no-sync, je-property, db-directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.
--add *property*:*value*. Adds a single value to a property, where *property* is the name of the property and *value* is the single value to be added.
--remove *property*:*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed. Valid property names are: all, global-index-deleted-entry-retention-timeout, db-cleaner-min-utilization, db-log-file-max, db-checkpointer-bytes-interval, db-checkpointer-wakeup-interval, db-num-lock-tables, db-num-cleaner-threads, db-txn-no-sync, db-txn-write-no-sync, je-property, db-directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.

set-index-prop
Modifies the properties of an index. Suboptions are as follows:
-c, --catalogName *name*. A unique identifier for the global index catalog. This is a required argument.
--attributeName attribute-name. The identifier for the global index attribute. This identifier should be unique in the context of the global index catalog and it is used to identify the global index.
--set *property*:*value*. Assigns a value to a property, where *property* is the name of the property and *value* is the single value to be assigned. Specify the same property multiple times to assign more than one value to it. Valid property names are: all, global-index-deleted-entry-retention-timeout, db-cleaner-min-utilization, db-log-file-max, db-checkpointer-bytes-interval, db-checkpointer-wakeup-interval, db-num-lock-tables, db-num-cleaner-threads, db-txn-no-sync, db-txn-write-no-sync, je-property, db-directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.
--reset *property*. Resets a property back to its default values, where *property* is the name of the property to be reset. Valid property names are: all, global-index-deleted-entry-retention-timeout, db-cleaner-min-utilization, db-log-file-max, db-checkpointer-bytes-interval, db-checkpointer-wakeup-interval, db-num-lock-tables, db-num-cleaner-threads, db-txn-no-sync, db-txn-write-no-sync, je-property, db-directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.
--remove *property*:*value*. Removes a single value from a property, where *property* is the name of the property and *value* is the single value to be removed. Valid property names are: all, global-index-deleted-entry-retention-timeout, db-cleaner-min-utilization, db-log-file-max, db-checkpointer-bytes-interval, db-checkpointer-wakeup-interval, db-num-lock-tables, db-num-cleaner-threads, db-txn-no-sync, db-txn-write-no-sync, je-property, db-directory, db-directory-permissions, global-index-catalogs-shared-cache, and global-index-attribute.

status-replication
Displays a list with the basic replication configuration of the global index catalog. If no global index catalog is specified, the information for all replicated global index catalogs is displayed. Suboptions are as follows:

`-c, --catalogName` *name*. A unique identifier for the global index catalog. This is a required argument.
`--adminUID` *adminUID*. User ID of the global administrator used to bind to the server. For the `status-replication` subcommand, if no global administrator was defined previously, the global administrator will be created using the provided data.
`-s, --scriptFriendly`. Use the script-friendly mode.

LDAP Connection Options

The `gicadm` command contacts the directory server over SSL through the administration connector (described in Managing Administration Traffic to the Server). These connection options are used to contact the directory server.

`-h, --hostname` *host*
Directory server hostname or IP address.

`-D, --bindDN` *bindDN*
DN to use to bind to the server.

`-j, --bindPasswordFile` *filename*
The full path to the file containing the bind password.

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-N, --certNickname` *nickname*
Use the certificate for SSL client authentication.

`-o, --saslOption`*name=value*
SASL bind option.

`-p, --port` *port*
Directory server administration port number.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust any certificate that the server presents. This option can be used for testing purposes, but for security reasons, a trust store should be used to determine whether the client should accept the server certificate.

`--connectTimeout` *timeout*
Specifies the maximum duration of time (in milliseconds) that can be taken to establish a connection. Use `0`to indicate no time out. The default value is 30000 milliseconds.

Command Input/Output Options

`--noPropertiesFile`
Indicate that the command will not use a properties file to get the default command-line options.

`--propertiesFilePath` ***propertiesFilePath***
Specify the path to the properties file that contains the default command-line options.

`-v, --verbose`
Run in verbose mode, displaying diagnostics on standard output.

General Options

`-?, -H, --help`
Displays command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

`-V, --version`
Displays the version information for the directory server.

Examples

The following examples show how to use the `gicadm` command.

> **Note:**
>
> The following examples for creating a global index catalog, adding a global index, and associating a global index catalog to a distribution are the three steps required to use a global index catalog in a distribution deployment.

Viewing the Global Help Subcommands and Global Options

The following command displays the available global Help subcommands and global options for managing the global index catalog:

```
$ gicadm --help
```

Viewing Help on an Individual Subcommand

The following command displays the help information for the `create-catalog` subcommand:

```
$ gicadm create-catalog --help
```

Using gicadm to Create a Global Index Catalog

You must have deployed the proxy with distribution before running this command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
create-catalog --catalogName myCatalog
```

Using gicadm to Add a Global Index to a Global Index Catalog

You must have deployed the proxy with distribution before running this command. Moreover, you must already have created the global index catalog before running this command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j /tmp-pwd-file -X \
add-index --catalogName myCatalog --attributeName telephoneNumber
```

Using gicadm to Associate a Global Index Catalog to a Distribution

You must have deployed the proxy with distribution before running this command. Moreover, you must already have created the global index catalog before running this command.

```
$ gicadm -h localhost -p 4444 -D "cn=Directory Manager" -j /tmp-pwd-file -X \
associate --catalogName myCatalog --distributionWorkflowElement myDistributionName
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/gicadm`
- Windows: *INSTANCE_DIR*`\OUD\bat\gicadm.bat`

Related Commands

- dsconfig
- split-ldif

A.1.2.9 manage-tasks

The `manage-tasks` command manages and monitors tasks that have been scheduled to run on the directory server.

This command is not supported for the proxy.

Synopsis

`manage-tasks` [*options*]

Description

The `manage-tasks` command can be used to manage and monitor tasks that have been scheduled to run on the directory server. Tasks are scheduled by providing the appropriate scheduling information when the task is invoked (see Configuring Commands As Tasks). The `manage-tasks` command can be used to list tasks that are currently scheduled or that have already been executed. In addition, you can get more detailed information about a task's scheduled and execution time, its log messages, and its options.

The `manage-tasks` command can only be run on an online server instance, and accesses the task back end over SSL through the administration connector (described in Managing Administration Traffic to the Server).

Options

The `manage-tasks` command accepts an option in either its short form (for example, `-c` *taskID*) or its long form equivalent (for example, `--cancel` *taskID*).

`-c, --cancel` *taskID*
Specify a particular task to cancel.

`-i, --info` *taskID*
Display information for a particular task.

`-s, --summary`
Print a summary of tasks.

LDAP Connection Options

`-D, --bindDN` *bindDN*
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` *hostname*
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` *filename*
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-N, --certNickname` *nickname*
Use the specified certificate for client authentication.

`-o, --saslOption` *name=value*
Use the specified options for SASL authentication.

`-p, --port` *port*
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Command Input/Output Options

-n,--no-prompt
Use non-interactive mode. If required option values are missing, you are not prompted and the command will fail.

--noPropertiesFile
Indicates that a properties file is not used to obtain the default command-line options.

--propertiesFilePath ***path***
Specify the path to the properties file that contains the default command-line options.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to manage tasks.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `manage-tasks` command.

Displaying a Summary of Scheduled Tasks

The following command displays a list of scheduled tasks:

```
$ manage-tasks -h localhost -p 4444 -D "cn=directory manager" -j /path/pwd-file \
  -X -s

  ID                Type     Status
  ------------------------------------------------
  2008101610361710  Backup   Completed successfully
  2008101610403710  Restore  Completed successfully
  2008101610442610  Restore  Waiting on start time
```

Obtaining Task Information

The following command returns information about a specific task:

```
$ manage-tasks -h localhost -p 4444 -D "cn=directory manager" -j /path/pwd-file \
  -X -i 2008101610442610
  Task Details
  ------------------------------------------------------
  ID                       2008101610442610
  Type                     Restore
  Status                   Waiting on start time
  Scheduled Start Time     Jan 25, 2009 12:15:00 PM SAST
  Actual Start Time
  Completion Time
  Dependencies             None
  Failed Dependency Action None
  Email Upon Completion    admin@example.com
  Email Upon Error         admin@example.com

  Restore Options
  ---------------------------------
  Backup Directory  /backup/userRoot
```

Canceling a Scheduled Task

The following command cancels a scheduled task. The command uses the `--no-prompt` option to run in non-interactive mode.

```
$ manage-tasks -h localhost -p 4444 -D "cn=directory manager" -j /path/pwd-file \
  -X -c 2008101610442610
Task 2008101610442610 canceled
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `manage-tasks` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

- UNIX and Linux: *OUD_ORACLE_HOME*`/bin/manage-tasks`
- Windows: *OUD_ORACLE_HOME*`\bat\manage-tasks.bat`

Related Commands

- import-ldif
- export-ldif
- backup
- restore
- stop-ds

A.1.2.10 oudCopyConfig

The `oudCopyConfig` command is used to obtain a copy of an existing configuration, from the source environment.

For more information about moving from a test to production environment, see Moving from a Test to a Production Environment .

Synopsis

`oudCopyConfig` [*options*]

Description

To obtain a copy of an existing configuration, run the `oudCopyConfig` command in the source environment.

The `oudCopyConfig` command performs the following actions:

- It creates an archive (`archivePath`) that contains the required configuration data to move the test instance (`instHomePath`) to a production environment. The `-archiveLoc` option specifies the full path to the archive.

- It creates a move plan in the archive.
- Logs any messages to *log_directory.* If not specified, the default location of logged messages is the system temporary directory.

Options

The `oudCopyConfig` command accepts an option in the form:

`-javaHome`, *javaHomePath*
Absolute path of JDK.

`-al, -archiveLoc` *archivePath*
Absolute path of archive location. It contains the required configuration data to move the test instance (`instHomePath`) to a production environment.

`-sih, -sourceInstanceHomeLoc` *instHomePath*
Absolute path of an existing instance that you want to copy to a production environment.

`-h, -help`
Show this help message and exit. This parameter is **optional.**

`-ldl, -logDirLoc` *logPath*
Existing log directory location. Default location is system temporary location. This parameter is **optional.**

Examples

The following examples show how to use the `oudCopyConfig` command.

Obtaining a Copy of an Existing Configuration

The following command obtains a copy of an existing configuration.

```
$ OUD_ORACLE_HOME/bin/oudCopyConfig -javaHome /usr/jdk \
-sourceInstanceHomeLoc /local/asinst_1 -archiveLoc /tmp/oud.jar \
-logDirLoc /tmp/logs
```

Running the Help Command Option

The following command runs the Help Command Option.

```
$ OUD_ORACLE_HOME/bin/oudCopyConfig -javaHome /usr/jdk -help
```

Location

- UNIX and Linux: *OUD_ORACLE_HOME*`/bin/oudCopyConfig`
- Windows: *OUD_ORACLE_HOME*`\bat\oudCopyConfig.bat`

Related Commands

- oudExtractMovePlan
- oudPasteConfig

A.1.2.11 oudExtractMovePlan

The `oudExtractMovePlan` command is used to create an editable version of the configuration in a file named *moveplan.xml,* in the location specifed by the `-planDirLoc` argument. This directory must exist, and be writable.

For more information about moving from a test to production environment, see Moving from a Test to a Production Environment .

Synopsis

`oudExtractMovePlan` [*options*]

Description

You can modify certain configuration parameters by editing the move plan. A move plan is an XML file that exposes customizable parameters during the move across environments.

The move plan is generated when you run the `oudCopyConfig` command and is used by the `oudPasteConfig` command to duplicate the configuration.

Options

The `oudExtractMovePlan` command accepts an option in the form:

`-javaHome`, *javaHomePath*
Absolute path of JDK.

`-al, -archiveLoc` *archivePath*
Absolute path of archive location.

`-pdl, -planDirLoc` *planPath*
Absolute path to directory where moveplan is to be extracted. The name of move plan file is *moveplan.xml.*

`-h, -help`
Show this help message and exit. This parameter is **optional.**

`-ldl, -logDirLoc` *logPath*
Existing log directory location. Default location is system temporary location. This parameter is **optional.**

Examples

The following examples show how to use the `oudExtractMovePlan` command.

Editing the Configuration

The following command allows you to edit the configuration.

```
$ OUD_ORACLE_HOME/bin/oudExtractMovePlan -javaHome /usr/jdk \
-al /tmp/oud.jar -pdl /tmp -logDirLoc /tmp/logs
```

Running the Help Command Option

The following command runs the Help Command Option.

```
$ OUD_ORACLE_HOME/bin/oudExtractMovePlan -javaHome /usr/jdk -help
```

Location

- UNIX and Linux: *OUD_ORACLE_HOME*`/bin/oudExtractMovePlan`
- Windows: *OUD_ORACLE_HOME*`\bat\oudExtractMovePlan.bat`

Related Commands

- oudCopyConfig
- oudPasteConfig

A.1.2.12 oudPasteConfig

The `oudPasteConfig` command is used to paste the configuration in the target environment.

For more information about moving from a test to production environment, see Moving from a Test to a Production Environment .

Synopsis

`oudPasteConfig` [*options*]

Description

To obtain the configuration in the target environment, run the `oudPasteConfig` command.

The `oudPasteConfig` command creates a new server instance with the configuration obtained from the archive and the amended move plan.

Options

The `oudPasteConfig` command accepts an option in the form:

`-javaHome`, *javaHomePath*
Absolute path of JDK.

`-al, -archiveLoc` *archivePath*
Absolute path of archive location.

`-mpl, -movePlanLoc` *planPath*
Absolute path to the moveplan extracted during extract plan operation.

`-tih, -targetInstanceHomeLoc` *instHomePath*
Absolute path of instance home under which Oracle Unified Directory configuration will be restored.

`-toh, -targetOracleHomeLoc` *oracleHomePath*
Absolute path of the Oracle home associated with the instance home.

`-tin, -targetInstanceName` *instanceName*
Target instance name. If specified, must be consistent with target instance path. This parameter is **optional.**

`-h, -help`
Show this help message and exit. This parameter is **optional.**

`-ldl, -logDirLoc` *logPath*
Existing log directory location. Default location is system temporary location. This parameter is **optional.**

Examples

The following examples show how to use the `oudPasteConfig` command.

Pasting the Configuration

The following command allows you to paste the configuration.

```
$ OUD_ORACLE_HOME/bin/oudPasteConfig -javaHome /usr/jdk -al /tmp/oud.jar \
-tih /tmp/asinst_2 -toh /tmp/Oracle_OUD1 \
-mpl /tmp/moveplan.xml -tin asinst_2
```

Running the Help Command Option

The following command runs the Help Command Option.

```
$ OUD_ORACLE_HOME/bin/oudPasteConfig -javaHome /usr/jdk -help
```

Location

- UNIX and Linux: *OUD_ORACLE_HOME*/`bin/oudPasteConfig`
- Windows: *OUD_ORACLE_HOME*`\bat\oudPasteConfig.bat`

Related Commands

- oudCopyConfig
- oudExtractMovePlan

A.1.2.13 oud-replication-gateway-setup

The `oud-replication-gateway-setup` command is used to setup the replication gateway instance.

Synopsis

`oud-replication-gateway-setup` [*options*]

Description

The `oud-replication-gateway-setup` command installs and configures a replication gateway instance, including specifying the ports on which it will listen, the DN and password for the initial root user, and the base DN for the replication gateway data. The replication gateway allows replication to work between a set of Oracle Directory Server Enterprise Edition servers and a set of Oracle Unified Directory servers.

The utility can be run in one of the following modes:

- **Graphical-user interface (GUI) mode**. GUI mode is the default and recommended installation option. The `oud-replication-gateway-setup` GUI provides an easy interface for installing and configuring replication servers in replicated multi-network environments. GUI mode also allows for easy server setup using SSL or StartTLS if desired.

  The utility launches the graphical installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on.

- **Command-line interface (CLI) mode**. The command-line mode is either interactive or non-interactive. The interactive CLI mode prompts you for any required information before the configuration begins, and is used with the `--cli` option, or if no GUI is available.

  The utility launches the command-line installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name

is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on.

The non-interactive CLI mode enables you to set up the server without user intervention. Use the `--no-prompt` and the `--quiet` options to suppress interactivity and output information, respectively.

When the `oud-replication-gateway-setup` command is run without any options, it starts in GUI mode but falls back to interactive command-line mode if no GUI is available. To run the setup in interactive command-line mode, use the `--cli` option.

> **Note:**
>
> No options are allowed if the command is run in GUI mode.

Options

The `oud-replication-gateway-setup` command accepts an option in either its short form (for example, `-i`) or its long form equivalent (for example, `--cli`).

`-i, --cli`
Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified.

Replication Gateway Configuration Options

`-h, --hostname` ***hostname***
The fully-qualified name of the host where the replication gateway will be installed. The Oracle Directory Server Enterprise Edition and Oracle Unified Directory servers in the replication topology must be able to access this hostname. If this option is not provided, a default of `localhost` is used.

`--adminConnectorPort` ***port***
Specifies the port on which the administration connector should listen for administration traffic. For information about the administration connector, see Managing Administration Traffic to the Server. The configuration and administration tools use this port to connect to the replication gateway. The default value is `4444`.

`--replicationPortForLegacy` ***port***
Specifies the port that is used by the Oracle Directory Server Enterprise Edition server to communicate with the replication gateway to replicate contents.

`-S, --skipPortCheck`
Do not make any attempt to determine whether the specified port is available. Normally, when this option is not present, the `oud-replication-gateway-setup` command verifies if that port is in use or not, and if not in use then the user running the command can bind to that port. With the `--skipPortCheck` option, the `oud-replication-gateway-setup` command skips the port check.

`-D, --rootUserDN` ***rootUserDN***
DN for the initial root user for the replication gateway.

`-j, --rootUserPasswordFile` *rootUserPasswordFile*
Path to a file containing the password for the initial root user for the replication gateway.

`-O, --doNotStart`
Do not start the replication gateway when the configuration is completed.

`-b, --baseDN` *baseDN*
Specify the base DN of the data to be replicated between the Oracle Unified Directory and the Oracle Directory Server Enterprise Edition server. Multiple base DN's can be provided by using this option multiple times.

Oracle Directory Server Enterprise Edition Server Options

`--hostNameLegacy` *hostname*
The fully-qualified name of the host or IP address of the Oracle Directory Server Enterprise Edition server whose contents will be replicated.

`--portLegacy` *port*
Specifies the port number of the Oracle Directory Server Enterprise Edition server whose contents will be replicated. This port is used by the replication mechanism to replicate contents.

`--bindDNLegacy` *bindDN*
Specifies the DN that is used to bind the Oracle Directory Server Enterprise Edition server whose contents will be replicated.

`--bindPasswordFileLegacy` *bindPasswordFile*
Specifies the file that stores the password that is used to bind the Oracle Directory Server Enterprise Edition server whose contents will be replicated.

`--secureReplicationLegacy`
Specifies if the replication updates between the Oracle Directory Server Enterprise Edition server and the replication gateway are sent encrypted or not. If you enable this option, then you must specify the certificate to be used by the server using the options in `Replication Gateway Security Options` and the port specified using argument `--portLegacy` must be an LDAP port.

`--clientAuthenticationToLegacy`
Uses client authentication to send replication updates from the replication gateway to the Oracle Directory Server Enterprise Edition server. You can use this argument only if attribute `--secureReplicationLegacy` is used.

`--certFileForClientAuthenticationToLegacy` *certificateFile*
Specifies the file that contains the certificate to be used in client authentication mode when the replication gateway connects to the Oracle Directory Server Enterprise Edition server to send replication updates. The file must contain the certificate in X.509 format.

`--doNotSendUpdatesToLegacyServer`
Do not propagate the updates made in the Oracle Unified Directory servers to the Oracle Directory Server Enterprise Edition server. If you use this option the changes made directly in the Oracle Unified Directory servers will not be propagated to the Oracle Directory Server Enterprise Edition servers replication topology.

`--doNotUpdateTrustStoreWithLegacyCertsArg`
If you specify this argument and the replication gateway sends replication updates to the Oracle Directory Server Enterprise Edition server using an encrypted communication

(specified using the `--secureReplicationLegacy` argument), then you will have to update the trust store used by the replication gateway with the server certificate of the Oracle Directory Server Enterprise Edition server for replication to work.

`--clientAuthenticationFromLegacy`
Uses client authentication to send replication updates from the Oracle Directory Server Enterprise Edition server to the replication gateway. You can use this argument only if attribute `--secureReplicationLegacy` is used.

Replication Gateway Security Options

`--generateSelfSignedCertificate`
Generates a self-signed certificate that the replication gateway will use as server certificate when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

`--usePkcs11Keystore`
Use a certificate in a PKCS#11 token that the replication gateway will use as server certificate when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

`--useJavaKeystore` ***keyStorePath***
Specifies the path of a Java Key Store (JKS) that contains a certificate that the replication gateway will use as server certificate when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

`--useJCEKS` ***keyStorePath***
Specifies the path of a JCEKS that contains a certificate that the replication gateway will use as server certificate when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

`--usePkcs12keyStore` ***keyStorePath***
Path of a PKCS#12 key store that contains the certificate that the replication gateway will use as server certificate when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

`--gatewayKeyStorePasswordFile` ***keyStorePasswordFile***
Specifies the file containing the certificate key store PIN. It is required to access the key store that contains the certificate (JKS, JCEKS, PKCS#12, or PKCS#11) that the replication gateway will use as server certificate. This is required when the replication gateway is configured for encrypted replication communication with the Oracle Directory Server Enterprise Edition server.

`--gatewayCertNickname` ***nickname***
Specifies the nickname of the certificate that the replication gateway will use when accepting encrypted connections from the Oracle Directory Server Enterprise Edition server.

Oracle Unified Directory Server Options

`--hostNameNg` ***hostname***
The fully-qualified name of the host or IP address of the Oracle Unified Directory server whose contents will be replicated.

`--portNg` ***port***
Specifies the port number of the Oracle Unified Directory server whose contents will be replicated.

`--bindDNNg` *bindDN*
Specifies the DN that is used to bind the Oracle Unified Directory server whose contents will be replicated. If this attribute is not specified the global administrator is used to bind.

`--bindPasswordFileNg` *bindPasswordFile*
Specifies the file that stores the password that is used to bind the Oracle Unified Directory server whose contents will be replicated. If no bind DN is specified for this server the password of the global administrator is used to bind.

`--replicationPortNg` *port*
Specifies the port used by the replication mechanism in the Oracle Unified Directory server to communicate with other Oracle Unified Directory servers. You must specify this option only if you have not configured replication for the provided Oracle Unified Directory server.

`--secureReplicationNg`
Specifies whether the replication communication established by the replication gateway to the Oracle Unified Directory servers is encrypted. If the replication port of the Oracle Unified Directory was not configured, the communication through it will be encrypted depending on whether this option is set.

`-I, --adminUID` *adminUID*
Specifies the user ID of the Global Administrator to use to bind to the Oracle Unified Directory server. If you have not defined a Global Administrator in the Oracle Unified Directory, then the Global Administrator is created using the provided data. The default value is admin.

`--adminPasswordFile` *bindPasswordFile*
The file that contains the password of the global administrator.

Secure Connection Options

`-o, --saslOption` *name=value*
These are SASL bind options.
SASL is not supported for a proxy instance.

`-X, --trustAll`
Trust all server SSL certificates that the server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-U, --trustStorePasswordFile` *path*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-N, --certNickname` *nickname*
Use the specified certificate for SSL client authentication.

`--connectTimeout` *timeout*
Specifies the maximum length of time (in milliseconds) that can be taken to establish a connection. Use `0`to specify no time out. The default value is 30000.

Command Input/Output Options

`-n, --no-prompt`
Run `setup` in non-interactive mode. If some data in the command is missing, the user will not be prompted and the command will fail.

`-Q, --quiet`
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

`-v, --verbose`
Run in verbose mode, displaying diagnostics on standard output.

`--noPropertiesFile`
Indicate that the command will not use a properties file to get the default command-line options.

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`--version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the replication server commands.

Running oud-replication-gateway-setup in GUI Mode

The following command runs an installation in GUI mode:

```
$ oud-replication-gateway-setup
```

The utility launches the graphical installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the *INSTANCE_NAME* environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-instance
```

Running oud-replication-gateway-setup in Interactive Mode From the Command Line

The GUI is launched and provides several screens that walk you through setting up your replication server in standalone or replicated environments. You also have the option to set up SSL or StartTLS certificates.

The `oud-replication-gateway-setup` command can be run in interactive mode, where you are prompted for installation options. To run `oud-replication-gateway-setup` in interactive mode, type the following command:

```
$ oud-replication-gateway-setup --cli
```

The command prompts you for the required setup values. Press Enter or Return to accept the default, or enter a value at the prompt.

The utility launches the command-line installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the *INSTANCE_NAME* environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-instance
```

Exit Codes

0
Successful completion or successful no-op.

1
Error unexpected. Potential bug.

2
Error user data. Cannot parse options, or data provided by user is not valid.

4
Error initializing server.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `oud-replication-gateway-setup` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

All the `oud-replication-gateway-setup` options can be stored in a properties file. Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
oud-replication-gateway-setup.hostname=grevalon:1444
```

Log Files

The `oud-replication-gateway-setup` command writes a log file named `oud-setup-`*IDnumber* where *IDnumber* is a decimal number. The log files are located at these paths:

- UNIX (Solaris): `/var/tmp/`
- Linux: `/tmp/`

- Windows: `%TEMP%`

  By default, this folder is `C:\Documents and Settings\User\Local Settings\Temp`.

Location

The `oud-replication-gateway-setup` command is located at these paths:

- UNIX and Linux: *OUD_BASE_LOCATION/OUD_ORACLE_HOME/*`oud-replication-gateway-setup`
- Windows: *OUD_BASE_LOCATION\OUD_ORACLE_HOME*`oud-replication-gateway-setup.bat`

Related Commands

- oud-setup
- oud-proxy-setup

A.1.2.14 oud-setup

The `oud-setup` command installs and minimally configures a directory server instance.

This command sets up a *directory server* instance. For information about setting up a proxy server instance, see oud-proxy-setup

Synopsis

`oud-setup` [*options*]

Description

The `oud-setup` command installs and configure a directory server instance, including specifying the ports on which it will listen, the DN and password for the initial root user, the base DN for the directory data, and the manner in which the database should be populated. It can be run in one of the following modes:

- **Graphical-user interface (GUI) mode**. GUI mode is the default and recommended installation option. The `oud-setup` GUI provides an easy interface for installing and configuring standalone directory servers or replication servers in replicated multi-network environments. GUI mode also allows for easy server setup using SSL or StartTLS if desired.

  The utility launches the graphical installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on.

- **Command-line interface (CLI) mode**. The command-line mode is either interactive or non-interactive. The interactive CLI mode prompts you for any required information before the configuration begins, and is used with the `--cli` option, or if no GUI is available.

  The utility launches the command-line installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on.

  The non-interactive CLI mode enables you to set up the server without user intervention. Use the `--no-prompt` and the `--quiet` options to suppress interactivity and output information, respectively.

When the `oud-setup` command is run without any options, it starts in GUI mode but falls back to interactive command-line mode if no GUI is available. To run `oud-setup` in command-line mode, use the `--cli` option. The options that can be provided are listed below.

> **Note:**
>
> No options are allowed if the command is run in GUI mode.

Options

The `oud-setup` command accepts an option in either its short form (for example, `-a`) or its long form equivalent (for example, `--addBaseEntry`).

`-a, --addBaseEntry`
Indicates whether to create the base entry in the directory server database.

`-i, --cli`
Run the `setup` command in command-line interactive mode rather than in GUI mode. If `setup` is run without the `--cli` option, it cannot accept other options.

`-b, --baseDN` ***baseDN***
Use the base DN for user information in the Directory Server. The default value for this option is `dc=example,dc=com`. Multiple base DNs can be specified by providing this option multiple times.

`-l, --ldifFile` ***filename***
Use the specified LDIF file to populate the database. Data can be imported from multiple files by providing this option multiple times, in which case the files are processed in the order they are provided in the option list. Do not use this option with either the `--addBaseEntry` or `--sampleData` option. If this option is not provided, then the database is left empty.

`-R, --rejectFile` ***filename***
Write rejected entries to the specified file. Rejected entries occur if they do not comply with the default schema during an import using the `-l` or `--ldifFile` option.

`--skipFile` ***filename***
Write skipped entries to the specified file. Skipped entries occur if entries cannot be placed under any specified base DN during an import using the `-l` or `--ldifFile` option.

`-d, --sampleData` ***number-of-entries***
Populate the database with the specified number of sample user entries. You generate the entries by using the MakeLDIF facility of the `import` command and they are based on the default `example.template` template. Do not use this option with either `--addBaseEntry` or `--ldifFile`. If this option is not provided, then the database is left empty.

`--eus`
Configure the server for Oracle's Enterprise User Security (EUS).

`-p,--ldapPort` ***port***
Contact the directory server at the specified port. If it is not provided, then the default port of `1389` as non-root and `389` as root is used. Use '*disabled*' if you do not want to enable it.

`--adminConnectorPort` *port*
Specifies the port on which the administration connector should listen for administration traffic. For information about the administration connector, see Managing Administration Traffic to the Server. The default value is `4444`.

`-x, --jmxPort` *port*
Specify the port for a JMX MBeans server connection. The default value for this option is `1689`.

`-S, --skipPortCheck`
Do not make any attempt to determine whether the specified port is available. Normally, when this option is not present, the `oud-setup` command verifies that the port is not in use and that the user running the setup command can bind to that port. With the `--skipPortCheck` option, the `oud-setup` command skips the port check.

`-D, --rootUserDN` *rootUserDN*
Use the specified root user DN to authenticate the directory server. This option is used when performing simple authentication and is not required if SASL authentication is used. The default value for this option is `cn=Directory Manager`.

`-j, --rootUserPasswordFile` *filename*
Specifies the file containing the password for the initial root user while authenticating the directory server.

`-O, --doNotStart`
Do not start the directory server when the configuration is completed.

`-q, --enableStartTLS`
Enable StartTLS to allow secure communication with the directory server by using the LDAP port.

`-Z, --ldapsPort` *port*
Contact the directory server at the specified port for LDAP SSL (LDAPS) communication. The LDAPS port will be configured and SSL will be enabled only if this option is explicitly specified. The default value is `1636`.

`--generateSelfSignedCertificate`
Generate a self-signed certificate that the directory server should use when accepting SSL-based connection or performing StartTLS negotiation.

`-h, --hostname` *host*
The name of the directory server host or IP address that is used to generate the self-signed certificate. This argument is considered only if the self-signed certificate argument, `--generateSelfSignedCertificate` is specified

`--usePkcs11Keystore`
Use a certificate in a PKCS#11 format that the server should use when accepting SSL-based connections or performing StartTLS negotiation

`--useJavaKeystore` *path*
Specify the path to the Java Keystore (JKS) that contains the server certificate.

`--useJCEKS` *path*
Specify the path to the Java Cryptography Extension Keystore (JCEKS) that contains the server certificate.

`--usePkcs12Keystore` *path*
Specify the path to the PKCS#12 keystore that contains the server certificate.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificate keystore. A password is required when you specify an existing certificate (JKS, JCEKS, PKCS#11, or PKCS#12) as a server certificate.

`-N, --certNickname` *nickname*
Use the specified certificate for SSL or StartTLS client authentication.

`-e, --enableWindowsService`
Enable the directory server as a Windows service. For Windows-platforms only.

`--serverTuning { jvm-default | heap-size | system-memory | system-memory-percentage | JVM arguments }`
Specifies runtime tuning options for the server.

> **Note:**
>
> From Oracle Unified Directory 11g Release 2 (11.1.2.3) onward, the `autotune` option has been removed. However, `autotune` usage is still available for backward compatibility.

To use the default (or configured) Java Virtual Machine with no extra arguments on your system when running the server, provide the value `jvm-default`.
To tune the server based on the heap size of its Java process, provide the memory to be used (for example: `768m`). For gigabytes, use `g` (for example: `2.5g`). If neither a unit nor a `%` is specified after the value, megabytes will be used.
The server requires memory not only for its Java process but also memory in the file-system cache. The memory for the server is the sum of the Java Heap and an estimation of the required file-system cache.
You can specify the system memory as either an amount or percentage:

- Amount: For example:

  Specify `--serverTuning systemMemory:3g` to use 3.0 gigabytes.

  Specify `--serverTuning systemMemory:1g` to use one gigabyte.

  Specify `--serverTuning systemMemory:512` to use 512 megabytes.

  The `oud-setup` script then splits the value you provide into two parts: the heap size that the Java Virtual Machine of the server will use and an estimation of the required file-system cache.

- Percentage: For example:

  Specify `--serverTuning systemMemory:50.0%` to dedicate 50 percent of system memory to the server.

  Specify `--serverTuning systemMemory:25%` to dedicate 25 percent of system memory to the server.

  Specify `--serverTuning systemMemory:100%` to fully dedicate a machine to the server.

The `oud-setup` script then splits the percentage you provide into two parts: the heap size that the Java Virtual Machine of the server will use and an estimation of the required file-system cache.

You can also directly specify the JVM arguments that the server must use. For example:

```
--serverTuning -server -Xmx1024m
```

The default value for the server will be calculated based on the free memory available on the system and will depend on the machine where the setup is running and how much memory is being used on that machine.

`--offlineToolsTuning { autotune | jvm-default | JVM arguments }`
Specifies tuning for the off-line tools (`import-ldif`, `export-ldif`, `verify-index`, and `rebuild-index`).

> **Note:**
>
> From Oracle Unified Directory 11g Release 2 (11.1.2.3) onward, the `--importTuning` option has been renamed to `--offlineToolsTuning`. However, `--importTuning` usage is still available for backward compatibility.

The tools can be automatically tuned each time they are launched based on the available memory in the machine (provide the value `autotune`), or they can use the default Java Virtual Machine on your system for the run-time settings (provide the value `jvm-default`).
You can also directly provide the Java arguments that the tools should use. For example, the following command uses the `-server` argument:

```
--offlineToolsTuning  -server -Xmx1024m
```

The default value for the tools will be calculated based on the free memory available on the system and will depend on the machine where the setup is running and how much memory is being used on that machine.

Command Input/Output Options

`-n, --no-prompt`
Run `setup` in non-interactive mode. If some data in the command is missing, the user will not be prompted and the command will fail.

`--noPropertiesFile`
Indicate that the command will not use a properties file to get the default command-line options.

`--propertiesFilePath` ***path***
Specify the path to the properties file that contains the default command-line options.

`-Q, --quiet`
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

`-v, --verbose`
Run in verbose mode, displaying diagnostics on standard output.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the directory server commands.

Running oud-setup in GUI Mode

The following command runs an installation in GUI mode:

```
$ oud-setup
```

The GUI is launched and provides several screens that walk you through setting up your directory server in standalone or replicated environments. You also have the option to set up SSL or StartTLS certificates.

The utility creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION* / *INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the INSTANCE_NAME environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-instance
```

Running oud-setup in Interactive Mode From the Command Line

The `oud-setup` command can be run in interactive mode, where you are prompted for installation options. To run `oud-setup` in interactive mode, type the following command:

```
$ oud-setup --cli
```

The command prompts you for the required setup values. Press Enter or Return to accept the default, or enter a value at the prompt.

The utility launches the command-line installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the *INSTANCE_NAME* environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-instance
```

Running oud-setup in Non-Interactive CLI Mode

The non-interactive CLI mode enables you to create installation scripts with the `oud-setup` command when many directory server instances must be configured for large replicated environments. This mode requires the `--no-prompt` and `--quiet` options to be provided. If no option is present, the `oud-setup` command defaults to interactive mode.

The following command runs the installation in non-interactive (`--no-prompt`) and quiet (`-Q`) modes. It sets the LDAP port (`-p`), the administration connector port (`--adminConnectorPort`),

the root DN (-D), the file containing the root DN password (-j), and adds a base entry (-a) with the specified base DN (-b),

```
$ oud-setup --cli --no-prompt -Q -p 1389 --adminConnectorPort 4444 \
  -D "cn=Directory Manager" -j /path/pwd-file -a -b dc=example,dc=com
```

Running oud-setup in Non-Interactive CLI Mode With LDIF Import

The following command runs the installation in non-interactive (--no-prompt) and quiet (-Q) modes. It sets the LDAP port (-p), the administration connector port (--adminConnectorPort), the root DN (-D), the file containing the root DN password (-j), and adds the baseDN (-b) with data imported from an LDIF file (-l).

```
$ oud-setup --cli --no-prompt -Q -p 1389 --adminConnectorPort 4444 \
  -D "cn=Directory Manager" -j /path/pwd-file -b dc=example,dc=com \
  -l "/home/ldif/company.ldif"
```

Running oud-setup in Non-Interactive Mode With Sample Entry Generation

The following command runs the installation in non-interactive (--no-prompt) and quiet (-Q) modes. It sets the LDAP port (-p), the administration connector port (--adminConnectorPort), the root DN (-D), the file containing the root DN password (-j), the baseDN (-b) and generates 2000 sample entries (-d).

```
$ oud-setup --cli --no-prompt -Q -p 1389 --adminConnectorPort 4444 \
  -D "cn=Directory Manager" -j /path/pwd-file -b dc=example,dc=com -d 2000
```

Running oud-setup on Windows

The following command enables the directory server to run as a Windows service (-e). It sets the LDAP port (-p), the administration connector port (--adminConnectorPort), the JMX port (-x), the rootDN (-D), the file containing the root DN password (-j), and the baseDN (-b), and generates 10000 sample entries.

```
C:\> oud-setup.bat --cli -e -p 1389 --adminConnectorPort 4444 -x 1689 \
  -D "cn=Directory Manager" -j /path/pwd-file -b dc=example,dc=com -d 10000
```

The utility launches the graphical installer and creates the Oracle Unified Directory instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is asinst_1, with subsequent instances on the same server named asinst_2, asinst_3, and so on. To specify a different instance name, set the INSTANCE_NAME environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-instance
```

Running oud-setup in Interactive Mode To Tune the Server

The following command allows you to tune the Oracle Unified Directory server.

```
C:\OUD\OracleUnifiedDirectory> oud-setup.bat --cli
OUD Instance location successfully created - C:\OUD\OracleUnifiedDirectory\..\asinst_1"

Oracle Unified Directory 14.1.2.1.0
Please wait while the setup program initializes...

What would you like to use as the initial root user DN for the Directory
Server? [cn=Directory Manager]:
Please provide the password to use for the initial root user: password
Please re-enter the password for confirmation: password

On which port would you like the Directory Server to accept connections from
```

```
LDAP clients? [389]:

On which port would you like the Administration Connector to accept
connections? [4444]:

Do you want to create base DNs in the server? (yes / no) [yes]:

Provide the base DN for the directory data: [dc=example,dc=com]:
Options for populating the database:

    1)  Only create the base entry
    2)  Leave the database empty
    3)  Import data from an LDIF file
    4)  Load automatically-generated sample data

Enter choice [1]: 4
Please specify the number of user entries to generate: [2000]:

Do you want to enable SSL? (yes / no) [no]:

Do you want to enable Start TLS? (yes / no) [no]:

Enable the server to run as a Windows Service? (yes / no) [no]:

Specify the Oracle components with which the server integrates.  It is
recommended to choose the option covering only your requirements.

    1)  No Integration
    2)  DIP (Directory Integration Platform)
    3)  Generic: Database Net Services, EBS and DIP
    4)  EUS (Enterprise User Security), Database Net Services, EBS and DIP

    c)  cancel

Enter choice [1]:

How do you want the OUD server to be tuned?

    1)  Use specific Java Virtual Machine arguments
    2)  Use the default Java Virtual Machine settings
    3)  Provide the Java heap size to be used by the server
    4)  Provide the percentage of system memory to be used by the server
    5)  Provide the size of system memory to be used by the server

Enter choice [2]: 2

How do you want the off-line tools (import-ldif, export-ldif, verify-index and
rebuild-index) to be tuned?

    1)  Use specific Java Virtual Machine arguments
    2)  Use the default Java Virtual Machine settings
    3)  Automatic Tuning
    4)  Provide the Java heap size to be used by the off-line tools

Enter choice [2]: 3

Do you want to start the server when the configuration is completed? (yes /
no) [yes]:


Setup Summary
=============
```

```
LDAP Listener Port:                  389
Administration Connector Port:       4444
LDAP Secure Access:                  disabled
Root User DN:                        cn=Directory Manager
Directory Data:                      Create New Base DN dc=example,dc=com
                                     Base DN Data: Import Automatically-Generated
                                     Data (2000 Entries)
Integration with Oracle components:  No Integration
Server Runtime Settings:             Use the default Java Virtual Machine settings
Off-line Tools Runtime Settings:     Use Automatic Tuning

Start Server when the configuration is completed
Do not enable the server to run as a Windows Service


What would you like to do?

    1)  Set up the server with the parameters above
    2)  Provide the setup parameters again
    3)  Print equivalent non-interactive command-line
    4)  Cancel and exit

Enter choice [1]:

See C:\OUD\asinst_1\OUD\logs\oud-setup for a detailed
log of this operation.

Configuring Directory Server ..... Done.
Importing Automatically-Generated Data (2000 Entries) ........ Done.
Starting Directory Server ....... Done.

To see basic server configuration status and configuration you can launch
C:\OUD\asinst_1\OUD\bat\status.bat
...
```

Exit Codes

0
Successful completion or successful no-op.

1
Error unexpected. Potential bug.

2
Error user data. Cannot parse options, or data provided by user is not valid.

4
Error initializing server.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `oud-setup` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `certNickname`
- `hostname`

- `keyStorePasswordFile`

All the preceding `oud-setup` options can be stored in a properties file. Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
oud-setup.hostname=grevalon:1444
```

Log Files

The `oud-setup` command writes a log file named `oud-setup-`*IDnumber* where *IDnumber* is a decimal number. The log files are located at these paths:

- UNIX (Solaris): `/var/tmp/`
- Linux: `/tmp/`
- Windows: `%TEMP%`

  By default, this folder is `C:\Documents and Settings\User\Local Settings\Temp`.

Location

The `oud-setup` command is located at these paths:

- UNIX and Linux: *OUD_BASE_LOCATION/OUD_ORACLE_HOME*`/oud-setup`
- Windows: *OUD_BASE_LOCATION\OUD_ORACLE_HOME*`\oud-setup.bat`

Related Commands

- oud-replication-gateway-setup
- oud-proxy-setup

A.1.2.15 oud-proxy-setup

The `oud-proxy-setup` command manages the setup and configuration of a proxy server instance.

Synopsis

`oud-proxy-setup` [*options*]

Description

The `oud-proxy-setup` command installs and configures a proxy server instance, including specifying the ports on which it will listen, the DN and password for the initial root user, authentication methods, as well load balancing, distribution, and a global index catalog, depending on the deployment chosen.

The `oud-proxy-setup` can only be launched once. It can be run in one of the following modes:

- **Graphical-user interface (GUI) mode**. GUI mode is the default and recommended installation option. The setup GUI provides an easy interface for defining and deploying the proxy instance.

  The utility launches the graphical installer and creates the proxy instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is

asinst_1, with subsequent instances on the same server named asinst_2, asinst_3, and so on.

- **Command-line interface (CLI) mode**. The command-line setup defines the proxy port, host name, and security configuration. If you specify the --cli option with oud-proxy-setup then you must provide the required values in the command line, else the default values are used. If you do not provide any value for a parameter that has no default value then the setup fails, and an error message is displayed.

  The utility launches the command-line installer and creates the proxy instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is asinst_1, with subsequent instances on the same server named asinst_2, asinst_3, and so on.

  The proxy setup CLI mode prompts the user to accept the license. Use the --no-prompt option to automatically accept the license.

Options

The oud-proxy-setup command accepts an option in either its short form (for example, -i) or its long form equivalent (for example, --cli).

-i, --cli
Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified.

-p, --ldapPort *port*
Port on which the Directory Server should listen for LDAP communication. The default value is 389.

--adminConnectorPort *port*
Port on which the Administration Connector should listen for communication. The default value is 4444.

-S, --skipPortCheck
Skip the check to determine whether the specified ports are usable.

-D, --rootUserDN *rootUserDN*
DN for the initial root user for the proxy server.

-j, --rootUserPasswordFile *rootUserPasswordFile*
Path to a file containing the password for the initial root user for the proxy server.

-q, --enableStartTLS
Enable StartTLS to allow secure communication with the server using the LDAP port.

-Z, --ldapsPort *port*
Port on which the Directory Server should listen for LDAP SSL (LDAPS) communication. The LDAPS port will be configured and SSL will be enabled only if this argument is explicitly specified. The default value is 636.

--generateSelfSignedCertificate
Generate a self-signed certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation.

`--usePkcs11keyStore` *keyStorePath*
Path of a PKCS#11 key store containing the certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation.

`--useJavaKeystore` *keyStorePath*
Path of a Java Key Store (JKS) containing a certificate to be used as the server certificate.

`--useJCEKS` *keyStorePath*
Path of a JCEKS containing a certificate to be used as the server certificate.

`--usePkcs12keyStore` *keyStorePath*
Path of a PKCS#12 key store containing the certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation.

`-u, --keyStorePasswordFile` *keyStorePasswordFile*
Certificate key store PIN file. A PIN is required when you specify to use an existing certificate (JKS, JCEKS, PKCS#12, or PKCS#11) as server certificate.

`-N, --certNickname` *nickname*
Nickname of the certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation.

`-O, --doNotStart`
Do not start the server when the configuration is completed.

Command Input/Output Commands

`-Q, --quiet`
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

`-v, --verbose`
Use verbose mode

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

`--noPropertiesFile`
Indicate that a properties file will not be used to get the default command-line options.

`-n, --no-prompt`
Perform an installation in non-interactive mode, for license acceptance only. If some data in the command is missing the user will not be prompted and the command will fail.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `oud-proxy-setup` command.

Running oud-proxy-setup in GUI Mode

The following command runs an installation in GUI mode:

```
$ oud-proxy-setup
```

The utility launches the graphical installer and creates the proxy instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the `INSTANCE_NAME` environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-proxy-instance
```

Running oud-proxy-setup in Non-Interactive CLI Mode

The non-interactive CLI mode enables you to create installation scripts with the setup command when many proxy server instances must be configured for large replicated environments. This mode requires the `--no-prompt` and `--quiet` options to be provided. If no option is present, the setup command defaults to interactive mode.

The following command runs the installation in non-interactive (`--no-prompt`) and quiet (`-Q`) modes. It sets the LDAP port (`-p`), the administration connector port (`--adminConnectorPort`), the root DN (`-D`), and the file containing the root DN password (`-j`).

```
$ oud-proxy-setup --cli --no-prompt -Q -p 1389 --adminConnectorPort 4444 \
  -D "cn=Directory Manager" -j /path/pwd-file
```

The utility launches the command-line installer and creates the proxy instance in *OUD_BASE_LOCATION/INSTANCE_DIR*. The default instance directory name is `asinst_1`, with subsequent instances on the same server named `asinst_2`, `asinst_3`, and so on. To specify a different instance name, set the `INSTANCE_NAME` environment variable before you run the setup, for example:

```
$ export INSTANCE_NAME=my-oud-proxy-instance
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Log Files

The `oud-proxy-setup` command writes a log file named `oud-proxy-setup.log`, once the setup in complete. The log file is located at these paths:

- UNIX (Solaris):`/var/tmp/`
- Linux:`/tmp/`
- Windows: The `%TEMP%` folder. By default, this folder is `C:\Documents and Settings\`*user*`\Local Settings\Temp`

Location

- UNIX and Linux: *OUD_BASE_LOCATION/OUD_ORACLE_HOME*`/oud-proxy-setup`
- Windows: *OUD_BASE_LOCATION\OUD_ORACLE_HOME*`\oud-proxy-setup.bat`

Related Commands

- oud-replication-gateway-setup
- stop-ds

A.1.2.16 start-ds

The `start-ds` command starts an installed server instance.

Synopsis

`start-ds` [*options*]

Description

The `start-ds` command is used to start the server and to provide general server information.

You can run `start-ds` without any options, which starts the server as a background process. In this case, the script will not exit until the server has either started successfully or has encountered an error that prevents it from starting.

On UNIX systems, the server will not start if it cannot log the process ID at *INSTANCE_DIR*`/logs/server.pid`. Ensure that the file is writable by the user account that the server uses.

Options

The `start-ds` command accepts an option in either its short form (for example, `-N`) or its long form equivalent (for example, `--nodetach`).

`-L, --useLastKnownGoodConfig`
Attempt to start using the configuration that was in place at the last successful startup (if it is available) rather than using the current active configuration.

`-N, --nodetach`
Start the server as a foreground process that does not detach from the terminal. When the server is running in this mode, it can be stopped by using the `stop-ds` command from another window, or by pressing `Control+C` in the terminal window in which the server is running.

`-s, --systemInfo`
Display general information about the system on which the server is installed, including the instance and installation paths, and then exit rather than attempting to start the server.

`-t, --timeout` ***seconds***
Wait no longer than the maximum time (in seconds) before the command returns. (The server continues the startup process, regardless). A value of `0` indicates an infinite timeout, which means that the command returns only when the server startup is completed. The default value is 60 seconds. This option cannot be used with the `-N`, `--nodetach` option.

Command Input/Output Options

`-Q, --quiet`
Run in quiet mode. No output is generated unless a significant error occurs during the process.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `start-ds` command.

Starting the Server

The following command starts the server:

```
$ start-ds
```

Starting the Server as a Foreground Process

The following command starts the server as a foreground process. You can stop the server by running the `stop-ds` command from another window or by pressing Control+C in the terminal window in which the server is running.

```
$ start-ds -N

[25/Jul/2007:10:39:17 -0500] category=CORE severity=NOTICE msgID=458887
msg=The Directory Server has started successfully
```

Exit Codes

Exit Code	Description
0	Server started successfully.
1	Check error. Generated from incompatible options.
98	Server already started.
99	Server must start as a detached process.
100	Server must start as a non-detached process.
101	Server must start as a Windows service.
102	Server must start as a detached process and it is being called from a Windows service.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/start-ds`
- Windows: *INSTANCE_DIR*`\OUD\bat\start-ds.bat`

Related Commands

- stop-ds

A.1.2.17 status

The `status` command displays basic server status information.

Synopsis

`status` [*options*]

Description

The `status` command can be used to display basic server information, such as the status of the server (started or stopped), the configured connection handlers, or the list of defined back ends and suffixes.

If the server is started, the `status` command connects to the server over SSL, through the administration connector.

For more information, see Managing Administration Traffic to the Server.

If the server is stopped, you must run this command as a user with file system access rights to read the configuration files (particularly the `config.ldif` file).

> **Note:**
>
> Certain monitoring data can only be displayed when the server is running (for example, the number of entries in a back end).

LDAP Connection Options

The `status` command contacts the server over SSL through the administration connector (described in Managing Administration Traffic to the Server). These connection options are used to contact the server.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the server.

`-K, --keyStorePath` ***path***
Use the client keystore certificate in the specified path.

`-N, --certNickname` ***nickname***
Use the specified certificate for client authentication.

`-o, --saslOption` ***name=value***
Use the specified options for SASL authentication.
SASL is not supported for a proxy server instance.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Command Input/Output Options

`-n, --no-prompt`
Use non-interactive mode. If some data in the command is missing, you are not prompted and the command will fail.

`--noPropertiesFile`
Indicate that the command should not use a properties file to get the default command-line options.

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

`-r, --refresh` *period*
When this argument is specified, the status command will display its contents periodically. Used to specify the period (in seconds) between two displays of the status.

`-s, --script-friendly`
Run in "script friendly" mode. Display the output in a format that can be easily parsed by a script.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `status` command.

Displaying the Server Status

The following example displays the current status of a standalone server that is currently online:

```
$ status -D "cn=directory manager" -j /path/pwd-file -X -n

          --- Server Status ---
Server Run Status:        Started
Open Connections:         1

          --- Server Details ---
Host Name:                hostname
Administrative Users:     cn=Directory Manager
Installation Path:        /path/OracleUnifiedDirectory
Instance Path:            /path/asinst_1/OUD
Version:                  Oracle Unified Directory 11.1.2.3.0
Java Version:             1.7.0_67
Administration Connector: Port 4444 (LDAPS)

          --- Connection Handlers ---
Address:Port : Protocol    : State
-------------:-------------:---------
--           : LDIF        : Disabled
8989         : Replication : Enabled
0.0.0.0:161  : SNMP        : Disabled
0.0.0.0:636  : LDAPS       : Disabled
0.0.0.0:1389 : LDAP        : Enabled
0.0.0.0:1689 : JMX         : Disabled

          --- Data Sources ---
Base DN:                      dc=example,dc=com
Backend ID:                   userRoot
Entries:                      7
Replication:                  Enabled
Missing Changes:              0
Age Of Oldest Missing Change: not available
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

How to Use a Properties File

The server supports the use of a *properties file* that passes in any default option values used with the `status` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `bindDN`
- `bindPasswordFile`
- `certNickname`
- `hostname`
- `keyStorePasswordFile`
- `keyStorePath`
- `port`
- `saslOption`

  SASL is not supported for a proxy server instance.

- `trustAll`
- `trustStorePasswordFile`
- `trustStorePath`

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
status.bindPasswordFile=/path/pwd-file
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/status`
- Windows: *INSTANCE_DIR*`\OUD\bat\status.bat`

A.1.2.18 stop-ds

The `stop-ds` command stops a server instance.

Synopsis

`stop-ds` [*options*]

Description

The `stop-ds` command is used to stop or restart the server. It can operate on either a local or remote server instance.

The ability to perform a local stop of the server is currently only available on UNIX based systems. When run locally, `stop-ds` sends a kill signal to the server process. This method of stopping the server is used if `stop-ds` is run without any options and if a PID file (*INSTANCE_DIR*`/OUD/logs/server.pid`) exists.

The remote shutdown mechanism issues an LDAP request to create a task entry in the server. The command can be run from any system that can communicate with the server (local or remote). It can also be used to restart the server. In this case, the server does an "in-core" restart, which reinitializes itself without shutting down the JVM.

When it is run remotely, `stop-ds` communicates with the server over SSL, through the administration connector. For more information, see Managing Administration Traffic to the Server.

Options

The `stop-ds` command accepts an option in either its short form (for example, `-D` *bindDN*) or its long form equivalent (for example, `--bindDN` *bindDN*).

`-r,--stopReason` *reason*
Provide a human-readable reason for the shutdown. If a reason is provided, it appears in the server's error log, and is provided to shut down plugins and shut down listeners.

`-R,--restart`
Restart the server rather than shutting it down. If the `--restart` option is used along with authentication options, the server will reinitialize itself without shutting down the JVM. Because the JVM is not stopped, any configuration changes that require a JVM restart will not take

effect. If the `--restart` option is used without authenticating, the server will first stop, then start. A new process will replace the original server.

`-t,--stopTime` *time*
Indicates the date and time at which the shutdown operation begins as a server task, expressed in the format `YYYYMMDDhhmmss`. A value of `0` causes the shutdown to be scheduled for immediate execution. When this option is used, the operation is scheduled to start at the specified time, after which this command exits immediately.

`-Y,--proxyAs` *authzID*
Use authorization control during the shutdown request. The value provided for this option should be an authorization ID, which can be in the form `dn:` followed by a user DN or `u:` followed by a user name. Clients will use the proxy authorization v2 control as described in RFC 4370 (`http://www.ietf.org/rfc/rfc4370.txt`).

LDAP Connection Options

The `stop-ds` command contacts the server over SSL through the administration connector (described in Managing Administration Traffic to the Server). These connection options are used to contact the server.

`-D, --bindDN` *bindDN*
Use the bind DN to authenticate to the server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` *hostname*
Contact the server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile filename`
Use the bind password in the specified file when authenticating to the server.

`-K, --keyStorePath path`
Use the client keystore certificate in the specified path.

`-N, --certNickname nickname`
Use the specified certificate for client authentication.

`-o, --saslOption` *name=value*
Use the specified options for SASL authentication.
SASL is not supported for a proxy server instance.

`-p, --port` *port*
Contact the server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

-U, --trustStorePasswordFile *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

-X, --trustAll
Trust all server SSL certificates that the server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Command Input/Output Options

--noPropertiesFile
Indicate that a properties file will not be used to get the default command-line options.

--propertiesFilePath *path*
Specify the path to the properties file that contains the default command-line options.

-Q, --quiet
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

--version
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `stop-ds` command.

Stopping a Server Locally

The following command stops the server:

```
$ stop-ds
```

Stopping a Server Remotely

The following command stops a remote server instance.

```
$ stop-ds -h remotehost -p 4444 -D "cn=directory manager" -j /path/pwd-file -X
```

Restarting a Server Remotely

The following command restarts a remote server instance.

```
$ stop-ds -R -h remotehost -p 4444 -D "cn=directory manager" -j /path/pwd-file -X
```

Exit Codes

Exit Code	Description
0	Server stopped successfully.

Exit Code	Description
98	Server already stopped.
99	Server must be started.
100	Server must be stopped using a system call.
101	Server must be restarted using a system call.
102	Server must be stopped using a protocol.
103	Server must be stopped as a Windows service.
104	Server must be restarted as a Windows service.

How to Use a Properties File

The server supports the use of a *properties file* that passes in any default option values used with the `stop-ds` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications.

For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `bindDN`
- `bindPasswordFile`
- `certNickname`
- `hostname`
- `keyStorePasswordFile`
- `keyStorePath`
- `saslOption`

  SASL is not supported for a proxy server instance.
- `trustAll`
- `trustStorePasswordFile`
- `trustStorePath`

```
toolname.propertyname=propertyvalue
```

For example:

Entries in the properties file have the following format:

```
stop-ds.trustAll=yes
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/stop-ds`
- Windows: *INSTANCE_DIR*`\OUD\bat\stop-ds.bat`

Related Commands

- start-ds

A.1.2.19 uninstall

The `uninstall` command is used to uninstall the server instance. It is applicable for directory servers, proxy servers, and replication gateway servers. The command removes the server instance, and not the software.

Synopsis

`uninstall` [*options*]

Description

The `uninstall` command is used to uninstall a server instance. It can be run in one of the following modes:

- **Graphical-user interface (GUI) mode**. GUI mode is the default and recommended uninstallation option. The `uninstall` GUI provides an easy interface for removing instance files.
- **Command-line interface (CLI) mode**. The command-line mode is either interactive or non-interactive. The interactive CLI mode prompts you for any required information before the uninstallation begins, and is used with the `--cli` option, or if no GUI is available.

  The non-interactive CLI mode enables you to uninstall the instance files without user intervention. Use the `--no-prompt` and the `--quiet` options to suppress interactivity and output information, respectively.

Whether running in GUI mode or in command-line mode, uninstall lists the components that you can remove. If uninstall cannot remove all of the instance files, it displays a message that lists any directories that are still present.

Depending on the type of server installed, you are presented with different uninstall options. These are broadly categorized into the following:

- Options to Remove a Directory Server
- Options to Remove a Proxy Server
- Options to Remove a Replication Gateway Server

> **Note:**
>
> For any instance (directory server, proxy, or replication gateway) type that you decide to remove, the uninstall procedure also stops the server. In addition, for a server instance that is part of a replication topology, the uninstall procedure removes the server that is under deletion from that topology. On a Windows platform, if the instance was installed as a windows service, the windows service is unregistered.

Options to Remove a Directory Server

The `uninstall` command accepts an option in either its short form (for example, `-i`) or its long form equivalent (for example, `--cli`).

The basic options to remove a directory server instances are:

`-i, --cli`
Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified.

`-a, --remove-all`
Remove all components of the server (this option is not compatible with the rest of the remove options).

`-l, --server-libraries`
Remove server libraries and administrative tools.

`-d, --databases`
Remove all database content.

`-L, --log-files`
Remove all log files.

`-c, --configuration-files`
Remove configuration files.

`-b, --backup-files`
Remove all backup files.

`-e, --ldif-files`
Remove LDIF files.

`-f, --forceOnError`
Specifies whether the uninstall should continue if there is an error updating references to this server in remote server instances or not. This argument can only be used with the `--no-prompt` argument.

The following options apply to LDAP connections:

`-I, --adminUID` ***user-ID***
Specify the user ID of the global administrator to bind to the server.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the directory server.

`-o, --saslOption` ***name=value***
Use the specified options for SASL authentication.

`-X, --trustAll`
Trust any certificate that the server presents. This option can be used for testing purposes, but for security reasons, a trust store should be used to determine whether the client should accept the server certificate.

`-P, --trustStorePath` ***path***
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-U, --trustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-N, --certNickname` *nickname*
Use the certificate for SSL client authentication.

`--connectTimeout` *timeout*
Maximum length of time that can be taken to establish a connect in milliseconds. Use `0` to specify no timeout. The default value is 30000.

`-h, --referencedHostName` *host*
Specify the name of this host (or IP address) as it is referenced in remote servers for replication.

Options to Remove a Proxy Server

The `uninstall` command accepts an option in either its short form (for example, `-i`) or its long form equivalent (for example, `--cli`).

The basic options to remove a proxy server instance are:

`-i, --cli`
Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified.

`-a, --remove-all`
Remove all components of the server (this option is not compatible with the rest of the remove options).

`-l, --server-libraries`
Remove server libraries and administrative tools.

`-L, --log-files`
Remove all log files.

`-c, --configuration-files`
Remove configuration files.

`-b, --backup-files`
Remove all backup files.

`-e, --ldif-files`
Remove LDIF files.

`-f, --forceOnError`
Specifies whether the uninstall should continue if there is an error updating references to this server in remote server instances or not. This argument can only be used with the `--no-prompt` argument.

The following options apply to LDAP connections:

`-I, --adminUID` *user-ID*
Specify the user ID of the global administrator to bind to the server.

`-j, --bindPasswordFile` *filename*
Use the bind password in the specified file when authenticating to the directory server.

`-o, --saslOption` *name=value*
Use the specified options for SASL authentication.

`-X, --trustAll`
Trust any certificate that the server presents. This option can be used for testing purposes, but for security reasons, a trust store should be used to determine whether the client should accept the server certificate.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-N, --certNickname` *nickname*
Use the certificate for SSL client authentication.

`--connectTimeout` *timeout*
Maximum length of time that can be taken to establish a connect in milliseconds. Use `0` to specify no timeout. The default value is 30000.

`-h, --referencedHostName` *host*
Specify the name of this host (or IP address) as it is referenced in remote servers for replication.

Options to Remove a Replication Gateway Server

The `uninstall` command accepts an option in either its short form (for example, `-i`) or its long form equivalent (for example, `--cli`).

The basic options to remove an instance of the replication gateway server are:

`-i, --cli`
Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified.

`-f, --forceOnError`
Specifies whether the uninstall should continue if there is an error updating references to this server in remote server instances or not. This argument can only be used with the `--no-prompt` argument.

The following option applies to gateway connections:

`-h, --hostname` ***hostname***
The fully-qualified name of the host where the replication gateway is installed. This name must be the one provided during the setup of the replication gateway.

The following options apply to Oracle Unified Directory Server connections:

`-I, --adminUID` ***adminUID***
User ID of the Global Administrator to use to bind to the Oracle Unified Directory server. If no Global Administrator was defined previously in the new generation server, then provide a Bind DN. The default value is `admin`.

`--adminPasswordFile` ***bindPasswordFile***
File containing the password of the Global Administrator (or of the bind DN) to use to bind to the Oracle Unified Directory server.

The following options apply to Oracle Directory Server Enterprise Edition connections:

`--bindDNLegacy` ***bindDN***
Specifies the DN that is used to bind theOracle Directory Server Enterprise Edition server whose contents whose contents are replicated through the replication gateway. The default value is `cn=Directory Manager`.

`--bindPasswordFileLegacy` ***bindPasswordFile***
Specifies the file that stores the password that is used to bind the Oracle Directory Server Enterprise Edition server whose contents are replicated through the replication gateway.

The following options apply to secure connections:

`-o, --saslOption` ***name=value***
These are SASL bind options.
SASL is not supported for a proxy server instance.

`-X, --trustAll`
Trust all server SSL certificates that the server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

`-P, --trustStorePath` ***path***
Use the trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-U, --trustStorePasswordFile` ***path***
Use the password in the specified file to access the certificates in the trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-K, --keyStorePath` ***path***
Use the keystore certificate in the specified path.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the keystore. This option is only required if `--keyStorePath` is used.

`-N, --certNickname` *nickname*
Use the specified certificate for SSL client authentication.

`--connectTimeout` *timeout*
Specifies the maximum length of time (in milliseconds) that can be taken to establish a connection. Use `0`to specify no time out. The default value is 30000.

Command Input/Output Options

`-n, --no-prompt`
Run `setup` in non-interactive mode. If some data in the command is missing, the user will not be prompted and the command will fail.

`-Q, --quiet`
Run in quiet mode. No output will be generated unless a significant error occurs during the process.

`-v, --verbose`
Run in verbose mode, displaying diagnostics on standard output.

`--noPropertiesFile`
Indicate that the command will not use a properties file to get the default command-line options.

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`--version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the server commands.

Uninstalling by Using the Graphical Uninstaller

The following command opens the Uninstaller GUI and prompts you to select the components that must be deleted:

```
$ uninstall
```

Uninstalling by Using the Command Line

The following command prompts you to indicate whether all components, or specific components, should be removed, and then runs the `uninstall` command. If the server is running, you are prompted to stop the server before continuing.

```
$ uninstall --cli
```

Uninstalling in Non-Interactive CLI Mode

This mode enables you to create an uninstallation script with the `uninstall` command. It requires the `--no-prompt` (`-n`) and `--quiet` (`-Q`) options to be provided. If no option is present, the `uninstall` command defaults to interactive mode. Both, `-n` and `-Q` options work in the CLI mode only.

The following command uninstalls all instance components in non-interactive CLI mode.

```
$ uninstall --cli -a -n -Q
```

Exit Codes

The following exit codes are applicable for a directory server and a proxy server:

0
Successful.

1
User canceled the operation.

2
User provided invalid data.

3
Error accessing file system (reading/writing).

5
Error during the configuration of the Directory Server.

7
Error starting the Oracle Unified Directory server.

8
Error stopping the Oracle Unified Directory server.

9
Error disabling the Windows service.

10
Application specific error.

11
Error invoking an Oracle Unified Directory tool.

12
Bug.

13
Java version non-compatible.

14
User provided invalid input.

50
Print Version.

51
Print Usage.

`100`
Return code for errors that are non-specified.

The following exit codes are applicable for a gateway server:

`0`
Successful uninstall.

`1`
Unexpected error (potential bug).

`2`
Cannot parse arguments or data provided by user is not valid.

`3`
The user canceled the uninstall.

`4`
Incompatible Java version.

`5`
Error initializing the replication gateway configuration (loading the admin framework classes, and so on).

`6`
Error stopping the replication gateway.

`7`
Error unconfiguring windows service.

`8`
Error input limit.

`9`
Error updating ADS Contents.

`10`
An error with the configuration of the legacy server. The base DN specified in the replica configuration is not a valid DN.

`11`
One of the specified legacy (Oracle Directory Server Enterprise Edition) servers is not compatible.

`12`
One of the specified new generation (Oracle Unified Directory based) servers is not compatible.

`13`
The user does not accept the certificate.

`14`
The user does not want to continue because there were issues loading the configuration of some servers.

`15`
An error with the configuration of the replication gateway.

16
The user overcame the maximum number of tries in interactive mode.

17
The user aborted the uninstall.

18
Error accessing file system (for instance deleting installation files).

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `uninstall` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `adminUID`
- `bindPasswordFile`
- `certNickname`
- `hostname`
- `keyStorePasswordFile`
- `keyStorePath`
- `saslOption`

  SASL is not supported for Oracle Unified Directory.
- `trustAll`
- `trustStorePasswordFile`
- `trustStorePath`

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
uninstall.bindPasswordFile=/path/pwd-file
```

Log Files

The `uninstall` command writes a log file named `oud-uninstall-`*IDnumber*, where `IDnumber` is a decimal number. The log files are located at these paths:

- UNIX (Solaris): `/var/tmp/`
- Linux: `/tmp/`
- Windows: The `%TEMP%` folder. By default, this folder is `C:\Documents and Settings\user\Local Settings\Temp`.

Location

The `uninstall` command is located at these paths:

- UNIX and Linux: *INSTANCE_DIR*`/OUD/uninstall`

- Windows: *INSTANCE_DIR*`\OUD\uninstall.bat`

Related Commands

- oud-replication-gateway-setup
- oud-setup

A.1.2.20 windows-service

The `windows-service` command manually enables or disables the server as a Windows service.

Synopsis

`windows-service` [*options*]

Description

The `windows-service` command can be used to manually enable (or disable) the server as a Windows service. Windows services are applications similar to UNIX daemons that run in the background and are not in direct control by the user.

Command Options

The `windows-service` command accepts an option in either its short form (for example, `-d`) or its long form equivalent (for example, `--disableService`):

`-c,--cleanupService` ***service-name***
Disable the service and clean up the Windows registry information associated with the provided service name.

`-d, --disableService`
Disable server as a Windows service.

`-e, --enableService`
Enable server as a Windows service.

`-s, --serviceState`
Display the state of the server as a Windows service.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `windows-service` command.

Enabling the Server as a Windows Service

The following command enables the server as a Windows service:

```
$ windows-service -e
```

Disabling the Server as a Windows Service

The following command disables the server as a Windows service:

```
$ windows-service -d
```

Displaying a Status

The following command displays a status of the server as a Windows service:

```
$ windows-service -s
```

Exit Codes

0
Server started/stopped successfully.

1
Service not found.

2
Server start error. Server already stopped

3
Server stop error.

Location

INSTANCE_DIR`\OUD\bat\windows-service.bat`

Related Commands

- oud-setup
- oud-proxy-setup
- oud-replication-gateway-setup

A.1.3 Data Administration Commands

You can review the different options and examples of each data administration command.

- backup
- base64
- dbtest
- encode-password
- export-ldif
- import-ldif
- ldif-diff
- ldifmodify
- ldifsearch
- list-backends
- make-ldif
- manage-account

- rebuild-index
- restore
- split-ldif
- verify-index
- purge-backup

A.1.3.1 backup

The `backup` command archives the contents of one or more directory server back ends.

Synopsis

`backup` [*options*]

Description

The `backup` command archives the contents of one or more directory server back ends. The command can perform this operation immediately or at a scheduled time. For more information, see Configuring Commands As Tasks.

The `backup` command can be run when the server is online or offline. If the backup is run while the server is online, the command contacts the server over SSL, through the administration connector, and registers a backup task. For more information about the administration connector, see Managing Administration Traffic to the Server.

Options

The `backup` command accepts an option in either its short form (for example, `-B` *backupID*) or its long form equivalent (for example, `--incrementalBaseID` *backupID*).

`-a, --backUpAll`
Back up all configured back ends. Do not use this option with `--backendID`.

`-A, --hash`
Generate a hash, or message digest, of the contents of the backup archive. The hash can be used as a checksum during the restore process to ensure that the backup has not been altered.

`-B, --incrementalBaseID` ***backupID***
Specify the backup ID for the existing backup against which to take an incremental backup. If this ID is not provided, the incremental backup is based on the latest incremental or full backup contained in the backup directory.

`-c, --compress`
Compress the contents of the backup archive. The compression algorithm used may vary based on the back end type.

`-d, --backupDirectory` ***path***
Write the backup files to the specified directory. If multiple back ends are archived, a subdirectory is created below this path for each back end. Otherwise, the backup files are placed directly in this directory. Multiple backups for the same back end can be placed in the same directory. If an incremental backup is to be performed, the backup directory must already contain at least one full backup. This is a required option.

For an online backup, the root for relative paths is the instance directory, and not the current working directory. For example, if you specify `-d bknov2011`, the backup files will be placed in `instance-dir/bknov2011`.

`-i, --incremental`
Perform an incremental backup rather than a full backup. An incremental backup includes only the data that has changed since a previous incremental or full backup. Thus, running an incremental backup can be notably faster than a full backup. When restoring an incremental backup, it is first necessary to restore the original full backup and then any intermediate incremental backups, which can make the restore process somewhat slower than restoring just a full backup. Some types of back ends might not support performing incremental backups. In this case, this option is ignored and a full backup is performed.

`-I, --backupID` ***backupID***
Specify an identifier to use for the backup. If this is not provided, a backup ID is generated, based on the current time. The backup ID must be unique among all backups in the provided backup directory.

`-n, --backendID` ***backendID***
Specify the ID of the back end to be saved. This option can be used multiple times in a single command to indicate that multiple back ends should be backed up. The available back ends in the server can be determined by using the `dsconfig list-backends` command.

`-s, --signHash`
Generate a signed hash. This provides even stronger assurance that neither the backup archive nor the hash of its contents have been altered. This option can only be used if a connection to an online directory server instance is present. In this case, you must specify the `--hostname`, `--port`, `--bindDN`, and `--bindPasswordFile` options of the online directory server that will generate a signed hash of the archive.

`-y, --encrypt`
Encrypt the contents of the backup archive. This option can only be used if a connection to an online server instance is present. In this case, you must specify the `--hostname`, `--port`, `--bindDN`, and `--bindPasswordFile` options of the online directory server that will encrypt the archive.

Task Back End Connection Options

Running an online backup requires access to the tasks back end. Access to the tasks back end is provided over SSL through the administration connector. These connection options are used when the backup runs online.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***hostname***
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` ***path***
Use the client keystore certificate in the specified path.

`-N, --certNickname` ***nickname***
Use the specified certificate for client authentication.

`-o, --saslOption` ***name=value***
Use the specified options for SASL authentication.

`-p, --port` ***port***
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` ***path***
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

These options are used when you specify that the backup should run as a scheduled task.

`--completionNotify` ***emailAddress***
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

`--dependency` ***taskId***
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

`--errorNotify` ***emailAddress***
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

`--failedDependencyAction` ***action***
Specify the action that this task will take if one of its dependent tasks fails. The value must be one of `PROCESS`, `CANCEL`, or `DISABLE`. If no value is specified, the default action is `CANCEL`.

`--recurringTask` ***schedulePattern***
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

`-t, --start` ***startTime***
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 schedules the task for immediate

execution. When this option is specified, the operation is scheduled to start at the specified time after which the command exits immediately.

Command Input/Output Options

`--noPropertiesFile`
Indicates that a properties file is not used to obtain the default command-line options.

`--propertiesFilePath` ***path***
Specify the path to the properties file that contains the default command-line options.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to back up data.

`-V, --version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the directory server commands.

Backing Up All Configured Back Ends

The following command archives all directory server back ends (`-a`), compresses them (`-c`), and saves them to a specified directory (`-d`).

```
$ backup -a -c -d /tmp/backup
```

Display the contents of the backup directory, to see the subdirectories for each back end:

```
$ ls /tmp/backup
schema    tasks     userRoot
```

Display the contents of a subdirectory, to see that the system assigned a backup ID based on the current time.

```
$ ls /tmp/backup/userRoot/
backup-userRoot-20081015151640Z  backup.info
```

You can assign your own unique backup ID by using the `-I` option. For example:

```
$ backup -a -c -d /tmp/backup -I October08
```

Display the contents of the `userRoot` subdirectory to see the assigned backup ID.

```
$ ls /tmp/backup/userRoot/
backup-userRoot-October08        backup.info
```

Backing Up a Specific Back End

Use the `-n` option to specify a back end to be backed up. The following command archives the `userRoot` back end only.

```
$ backup -n userRoot -d /tmp/backup
```

Running an Incremental Backup

The following command archives all directory server back ends (`-a`), using incremental backup (`-i`), compresses them (`-c`), and saves the data to a directory (`-d`).

```
$ backup -a -i -c -d /tmp/backup
```

Running an Incremental Backup on a Specific Back End

Use the `list-backends` command to display the current configured back ends.

```
$ list-backends
Backend ID     : Base DN
---------------:--------------------
adminRoot      : cn=admin data
ads-truststore : cn=ads-truststore
backup         : cn=backups
config         : cn=config
monitor        : cn=monitor
schema         : cn=schema
tasks          : cn=tasks
userRoot       : "dc=example,dc=com"
```

The following command runs an incremental backup (`-i`) on the `userRoot` back end (`-n`), compresses the backup (`-c`), and saves the data to a directory (`-d`).

```
$ backup -i -n userRoot -c -d /tmp/backup/userRoot
```

Running an Incremental Backup Against an Existing Backup

Assume that you have created two archived incremental backup files by using the `-I` or `--backupID` option and assigned the IDs `1234` and `4898` to the two files, respectively:

```
/tmp/backup/userRoot> ls
./     backup-userRoot-1234   backup.info
../    backup-userRoot-4898   backup.info.save
```

The following command runs an incremental backup (`-i`) on all configured back ends (`-a`) based on the backup ID `1234` (`-B`), assigns a backup ID of `5438` to the incremental backup, and saves the data to a directory (`-d`).

```
$ backup -a -i -B 1234 -I 5438 -d /tmp/backup
```

The contents of `backup.info` show that the latest incremental backup (`backup_id=5438`) has a dependency on `backup_id=1234`:

```
$ backend_dn=ds-cfg-backend-id=userRoot,cn=Backends,cn=config

backup_id=4898
backup_date=20070727202906Z
incremental=false
compressed=false
encrypted=false
signed_hash=VmBG/VkfMAMMPnR6M8b5kZil7FQ=
property.last_logfile_name=00000000.jdb
property.archive_file=backup-userRoot-4898
property.cipher_algorithm=AES/CBC/PKCS5Padding
property.mac_algorithm=HmacSHA1
property.last_logfile_size=490554

backup_id=1234
backup_date=20070727202934Z
incremental=false
compressed=false
```

```
encrypted=false
signed_hash=VmBG/VkfMAMMPnR6M8b5kZil7FQ=
property.last_logfile_name=00000000.jdb
property.archive_file=backup-userRoot-1234
property.cipher_algorithm=AES/CBC/PKCS5Padding
property.mac_algorithm=HmacSHA1
property.last_logfile_size=490554

backup_id=5438
backup_date=20070727203107Z
incremental=true
compressed=false
encrypted=false
dependency=1234
property.last_logfile_name=00000000.jdb
property.archive_file=backup-userRoot-5438
property.last_logfile_size=490554
```

Backing Up All Configured Back Ends with Encryption and Signed Hash

The directory server provides support for backup encryption (using `--encrypt`), hash generation (using `--hash`), and signed hash (using `--signHash`) to secure archived data. These options require a connection to an online server instance, over SSL through the administration connector. When you use these options, you must therefore specify the connection details, including the host, administration port, bind DN and bind password file. You must also specify the certificate details for the SSL connection.

The following command archives all directory server back ends (`-a`), compresses them (`-c`), generates a hash (`-A`), signs the hash (`-s`), encrypts the data while archiving the data (`-y`), assigns a back end ID of `123`, and saves the data to a directory (`-d`). The self signed certificate is trusted using the `-X` (`--trustAll`) option.

```
$ backup -h localhost -D "cn=Directory Manager" -j /path/pwd-file -p 4444 -X \
  -a -c -A -s -y -I 123 -d /tmp/backup
Backup task 2008101609295810 scheduled to start immediately
...
```

Scheduling a Backup

Scheduling a backup requires online access to the tasks back end. Access to this back end is provided over SSL through the administration connector. When you schedule a backup, you must therefore specify the connection details, including the host, administration port, bind DN and bind password file. You must also specify the certificate details for the SSL connection.

The following command schedules a backup of all components (`-a`) and writes it to the `/tmp/backups` directory (`-d`). The start time is specified with the `--start` option. The backup sends a completion notification and error notification to `admin@example.com`. The self signed certificate is trusted using the `-X` (`--trustAll`) option.

```
$ backup -h localhost -D "cn=Directory Manager" -j /path/pwd-file -p 4444 -X \
  -a -d /tmp/backups --start 20090124121500 --completionNotify admin@example.com \
  --errorNotify admin@example.com
Backup task 2007102914530410 scheduled to start Jan 24, 2009 12:15:00 PM SAST
```

You can view this scheduled task by using the `manage-tasks` command. For more information, see Configuring Commands As Tasks.

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `backup` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

The `backup` command is located at these paths:

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/backup`
- Windows: *INSTANCE_DIR*`\OUD\bat\backup.bat`

Related Commands

- restore
- list-backends
- manage-tasks

A.1.3.2 base64

The `base64` command encodes binary strings using the base64 encoding format.

Synopsis

`base64` *subcommand*[*options*]

Description

The `base64` command encodes binary strings into text representations using the base64 encoding format. Base64 encoding is often used in LDIF files to represent non-ASCII character strings. It is also frequently used to encode certificate contents or the output of message digests such as MD5 or SHA.

Subcommands

The following subcommands are used with the `base64` command.

`decode`
Decodes base64-encoded information into raw data. Suboptions are as follows:
`-d, --encodedData` *encoded-data*. Base64-encoded data to be decoded to raw data.
`-f, --encodedDataFile` *filename*. Path to the file that contains the base64-encoded data to be decoded.
`-o, --toRawFile` *filename*. Path to the file to which the raw data should be written.

`encode`
Encodes raw data to base64. Suboptions are as follows:
`-d, --rawData` *raw-data*. Raw data to be base64-encoded.

`-f, --rawDataFile` *filename*. Path to the file that contains the raw data to be base64-encoded.
`-o, --toEncodedFile` *filename*. Path to the file to which the base64-encoded data should be written.

Global Options

`-?, -H, --help`
Display usage information.

`-V, --version`
Display directory server version information.

Examples

The following examples show how to use the directory server commands.

Base64 Encoding a String

The following command base64-encodes the string `opends`.

```
$ base64 encode -d opends
b3BlbmRz
```

Base64 Encoding the Contents of a File

The following command base64-encodes the file (`-f`) and writes to an output file (`-o`).

```
$ base64 encode -f myrawdata -o myencodeddata
```

Decoding a Base64-Encoded String

The following command decodes a base64-encoded string.

```
$ base64 decode -d b3BlbmRz
opends
```

Decoding the Contents of a Base64-Encoded File

The following command decodes the file base64-encoded file (`-f`) and writes to an output file (`-o`).

```
$ base64 encode -f myencodeddata -o myoutput
```

Base64-Encoding and Decoding on Linux Systems

The following command encodes and decodes on Linux from the command-line. After you enter the clear-text string, press `Control-D` to signal the end of input on the command line.

```
$ base64 encode
hello world
<CTRL-D>
aGVsbGBqd29ybGQK

$ base64 decode
aGVsbG8gd29ybGQK
<CTRL-D>
hello world
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/base64`
- Windows: *INSTANCE_DIR*`\OUD\bat\base64.bat`

A.1.3.3 dbtest

The `dbtest` command debugs an Oracle Berkeley Java Edition (JE) back end.

Synopsis

`dbtest` *subcommands* [*options*]

Description

The `dbtest` command is used to debug an Oracle Berkeley Java Edition (JE) back end. The command lists the root, entry, database containers, and the status of indexes in the database. The command also provides a dump of the database for debugging purposes.

A *back end* is a repository for storing data on a directory server. The back end uses some type of database (DB) to store data and to maintain a set of indexes that allow the back end to locate the entries in the directory. The primary database for the directory server is the Berkeley Java Edition (JE) database, which organizes its data as a single collection of keyed records in B-tree form.

You can use the `dbtest` command to access the following information:

- **Root container**. Specifies the back end ID and the directory for the back end.
- **Entry container**. Specifies the base DN that the entry container stores on disk, the database prefix to use for the database names, and the number of entries in the database. Each base DN of a JE back end is given its own entry container.
- **Database container**. Specifies the database name, type, and JE database name for the specific back end ID.
- **Index Status**. Specifies the index name, type, status and associated JE database.

Currently, the `dbtest` command is a read-only command and cannot alter the database. The command can run in online or offline mode. However, running `dbtest` in online mode can take considerably longer than running it in offline mode.

Subcommands

`dump-database-container`
Dump records from the database container. Suboptions are as follows:
`-b, --baseDN baseDN`. Base DN of the entry container to debug. Required.
`-d, --databaseName databaseName`. The name of the database container to debug. Required.
`-k, --minKeyValue value`. Only show records with keys that should be ordered after the provided value using the comparator for the database container.
`-K, --maxKeyValue value`. Only show records with keys that should be ordered before the provided value using the comparator for the database container.

`-n, --backendID backendID`. ID of the local DB back end to debug. Required.
`-p, --skipDecode`. Skip decoding the local database to its appropriate types.
`-q, --statsOnly`. Display the statistics only, rather than the complete data.
`-s, --minDataSize size`. Only show records whose data is no smaller than the provided value.
`-S, --maxDataSize size`. Only show records whose data is no larger than the provided value.

`list-database-containers`
List the database containers for the entry container. Suboptions are as follows:
`-b, --baseDN baseDN`. Base DN of the entry container to debug. Required.
`-n, --backendID backendID`. ID of the local DB back end to debug. Required.

`list-entry-containers`
List the entry containers for a root container. Suboptions are as follows:
`-n, --backendID backendID`. ID of the local DB back end to debug. Required.

`list-index-status`
List the status of indexes in an entry container. Suboptions are as follows:
`-b, --baseDN` *baseDN*. Base DN of the entry container to debug. Required.
`-n, --backendID` *backendID*. ID of the local DB back end to debug. Required.

`list-root-containers`
List the root containers used by all local DB back ends.

Global Options

The `dbtest` command accepts an option in either its short form (for example, `-H`) or its long form equivalent (for example, `--help`).

`-?, -H, --help`
Display the usage information.

`-V, --version`
Display directory server version information.

Examples

The following examples show how to use the directory server commands.

Displaying the List of Root Containers

The following command lists the root containers used by all local DB back ends:

```
$ dbtest list-root-containers
Backend ID  Database Directory
------------------------------
userRoot    db

Total: 1
```

Displaying a List of Entry Containers

The following command displays the list of entry containers on the local DB back end:

```
$ dbtest list-entry-containers -n userRoot
Base DN            JE Database Prefix  Entry Count
-------------------------------------------------
dc=example,dc=com  dc_example_dc_com   102

Total: 1
```

Displaying a List of Database Containers

The following command displays the list of database containers on the local DB back end:

```
$ dbtest list-database-containers -b dc=example,dc=com -n userRoot
Database Name              Database  JE Database Name                          Entry Count
                           Type
-------------------------------------------------------------------------------------------
dn2id                      DN2ID     dc_example_dc_com_dn2id                   102
id2entry                   ID2Entry  dc_example_dc_com_id2entry                102
referral                   DN2URI    dc_example_dc_com_referral                0
id2children                Index     dc_example_dc_com_id2children             2
id2subtree                 Index     dc_example_dc_com_id2subtree              2
state                      State     dc_example_dc_com_state                   19
objectClass.equality       Index     dc_example_dc_com_objectClass.equality    6
givenName.equality         Index     dc_example_dc_com_givenName.equality      100
givenName.substring        Index     dc_example_dc_com_givenName.substring     396
member.equality            Index     dc_example_dc_com_member.equality         0
uid.equality               Index     dc_example_dc_com_uid.equality            100
cn.equality                Index     dc_example_dc_com_cn.equality             100
cn.substring               Index     dc_example_dc_com_cn.substring            1137
uniqueMember.equality      Index     dc_example_dc_com_uniqueMember.equality   0
telephoneNumber.equality   Index     dc_example_dc_com_telephoneNumber.equality 100
telephoneNumber.substring  Index     dc_example_dc_com_telephoneNumber.substring 956
sn.equality                Index     dc_example_dc_com_sn.equality             100
sn.substring               Index     dc_example_dc_com_sn.substring            541
ds-sync-hist.ordering      Index     dc_example_dc_com_ds-sync-hist.ordering   0
mail.equality              Index     dc_example_dc_com_mail.equality           100
mail.substring             Index     dc_example_dc_com_mail.substring          525
entryUUID.equality         Index     dc_example_dc_com_entryUUID.equality      102
aci.presence               Index     dc_example_dc_com_aci.presence            0

Total: 23
```

Dumping the Contents of a Database and Skipping Decode

The following command dumps the contents of a database and displays the indexed values of the entry, but skips the decode.

```
$ dbtest dump-database-container -b dc=example,dc=com -n userRoot \
  -d objectClass.equality -p

Key (6 bytes):
64 6F 6D 61 69 6E domain

Data (8 bytes):
00 00 00 00 00 00 00 01

Key (18 bytes):
67 72 6F 75 70 6F 66 75 6E 69 71 75 65 6E 61 6D groupofu niquenam
65 73 es

Data (40 bytes):
00 00 00 00 00 00 00 03 00 00 00 00 00 00 00 9C
00 00 00 00 00 00 00 9D 00 00 00 00 00 00 00 9E
00 00 00 00 00 00 00 9F
...
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/dbtest`
- Windows: *INSTANCE_DIR*`\OUD\bat\dbtest.bat`

Related Commands

- dsconfig
- import-ldif
- export-ldif

A.1.3.4 encode-password

The `encode-password` command encodes and compares user passwords.

This command is not supported for the proxy.

Synopsis

`encode-password` *options*

Description

The `encode-password` command can be used to interact with the password storage schemes defined in the directory server. It has three modes of operation:

- **List schemes mode**. List the password storage schemes that are available in the directory server. In this mode, only the `--listSchemes` option is required.
- **Encode clear-text mode**. Encode a clear-text password using a provided password storage scheme. In this mode, the `--storageScheme` option is required, along with a clear-text password that is read from a file (`--clearPasswordFile`).
- **Validate password mode**. Determine whether a given clear-text password is correct for a provided encoded password. In this mode, a clear-text password (from `--clearPasswordFile`) and an encoded password (from `--encodedPasswordFile)` are required.

The set of authentication passwords available for use in the directory server can be retrieved from the `supportedAuthPasswordSchemes` attribute of the root DSE entry. You can use `ldapsearch` to view this information.

Options

The `encode-password` command accepts an option in either its short form (for example, `-f` *filename*) or its long form equivalent (for example, `--clearPasswordFile` *filename*).

`-a, --authPasswordSyntax`
Use the Authentication Password Syntax (as defined in RFC 3112 (`http://www.ietf.org/rfc/rfc3112.txt`)), which encodes values in a form *scheme*`$`*authInfo*`$`*authValue*. If this option is not provided, then the user password syntax (which encodes values in a form *scheme*`$`*value* will be used.

`-E, --encodedPasswordFile` *filename*
Use the encoded password from the specified file to compare against a given clear-text password. If the `--authPasswordSyntax` option is also provided, then this password must be encoded using the authentication password syntax. Otherwise, it should be encoded using the user password syntax.

`-f, --clearPasswordFile` *filename*
Use the clear-text password from the specified file when either encoding a clear-text password or comparing a clear-text password against an encoded password.

`-i, --interactivePassword`
The password to encode or to compare against an encoded password is interactively requested from the user.

`-l, --listSchemes`
Display a list of the password storage schemes that are available for use in the directory server. If the option is used by itself, it displays the names of the password storage schemes that support the user password syntax. If the option used with `--authPasswordSyntax`, then it displays the names of the password storage schemes that support the authentication password syntax.

`-r, --useCompareResultCode`
Use an exit code that indicates whether a given clear-text password matched a provided encoded password. If this option is provided, the directory server results in an exit code of `6` (`COMPARE_TRUE`) or an exit code of `5` (`COMPARE_FALSE`). Any other exit code indicates that the command failed to complete its processing to make the necessary determination. If this option is not provided, an exit code of zero will be used to indicate that the command completed its processing successfully, or something other than zero if an error occurred.

`-s, --storageScheme` *storageScheme*
Specify the name of the password storage scheme to use when encoding a clear-text password. If the `--authPasswordSyntax` option is provided, the value must be the name of a supported authentication password storage scheme. Otherwise, specify the name of a supported user password storage scheme.

`-?, -H, --help`
Display the command-line usage information for the command and exit immediately without taking any other action.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the `encode-password` command.

Listing the Storage Schemes on the Server

The following command lists the storage schemes (`-l`) available for use on the directory server.

```
$ encode-password -l
3DES
AES
BASE64
BLOWFISH
CLEAR
CRYPT
MD5
```

```
RC4
SHA
SMD5
SSHA
SSHA256
SSHA384
SSHA512
```

Listing the Authenticated Passcode Syntax Storage Schemes on the Server

The following command lists the storage schemes (`-l`) that support the authentication passcode syntax (`-a`) on the directory server.

```
$ encode-password -l -a

MD5
SHA1
SHA256
SHA384
SHA512
```

Encoding a Clear-Text Password to Another Scheme

The following command encodes a clear-text password in a file (`-f`) using the specified scheme (`-s`).

```
$ encode-password -f /path/clear-pwd-file -s MD5

Encoded Password:  "{MD5}AjxHKRFkRwxx3j9lM2HMow=="
```

Encoding a Clear-Text Password to Another Scheme using the Authentication Password Syntax

The following command encodes a clear-text password in a file (`-f`) using the specified scheme (`-s`) and the authentication password syntax (`-a`).

```
$ encode-password -f /path/clear-pwd-file -s MD5 -a

Encoded Password:  "MD5$/imERhcEu3U=$AFqmpZi8EiTIvMFwkcrf8A=="
```

Comparing a Clear-Text Password to an Encoded Password

The following command compares a clear-text password in a file (`-f`) with an encoded password in a file (`-E`). Do not include the password scheme (for example, `MD5`) in your encoded password.

```
$ encode-password -f /path/clear-pwd-file -E /path/encoded-pwd-file -s MD5

The provided clear-text and encoded passwords match
```

Compare a Clear-Text Password to an Encoded Password and Return an Exit Code

The following command compares a clear-text password in a file (`-f`) with an encoded password in a file (`-E`) using the scheme (`-s`) and returns the exit code (`-r`) (`6` for COMPARETRUE; `5` for COMPAREFALSE). Do not include the password scheme (for example, `MD5`) in your encoded password.

```
$ encode-password -f /path/clear-pwd-file -E /path/encoded-pwd-file -s MD5 -r

The provided clear-text and encoded passwords match
```

```
echo $?
6
```

Encoding a Password Contained in a File using SSHA

The following command encodes a clear-text password in a file (`-f`) using the specified scheme (`-s`). For Windows platforms, specify the path to your clear-text password file (for example, `-f \temp\password`):

```
$ encode-password -s SSHA -f /path/clear-pwd-file

Encoded Password:   "{SSHA}QX2fMu+2N22N9qI+zu6fIZxsBVID3EsUlYYEbQ=="
```

Exit Codes

Table A-3 Exit Codes

Exit Code	Description
0	Operation completed successfully.
1	Error occurred during operation.
5	COMPARE_FALSE. Used with the `--r` or `--useCompareCodeResult` option, an exit code of `5` indicates a given clear-text password does not match the provided encoded password.
6	COMPARE_TRUE. Used with the `--r` or `--useCompareCodeResult` option, an exit code of `6` indicates that a given clear-text password matches the provided encoded password.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/encode-password`
- Windows: *INSTANCE_DIR*`\OUD\bat\encode-password.bat`

A.1.3.5 export-ldif

The `export-ldif` command exports the contents of a directory server back end to LDIF format.

Synopsis

`export-ldif` [*options*]

Description

The `export-ldif` command exports the contents of a directory server back end to LDIF format. This command can run the export immediately or can be scheduled to run at a specified date and time. For more information, see Configuring Commands As Tasks.

Because some back ends cannot be imported to the directory server, the `export-ldif` command does not export the following back ends: `monitor`, `ads-truststore`, `backup`, and `config-file-handler`.

You can run the `export-ldif` command in online or offline mode.

- **Online mode**. In online mode, `export-ldif` contacts a running directory server instance over SSL, through the administration connector, and registers an export task. The command runs in online mode automatically if you specify any of the task back end

connection options. For more information about the administration connector, see Managing Administration Traffic to the Server.

- **Offline mode**. In offline mode, `export-ldif` accesses the database directly rather than through a directory server instance. To perform an offline export, the directory server must be stopped.

Options

The `export-ldif` command accepts an option in either its short form (for example, `-b` *branchDN*) or its long form equivalent (for example, `--includeBranch` *branchDN*).

`-a, --appendToLDIF`
Append the export to an existing LDIF file rather than overwriting it. If this option is not provided, the directory server overwrites the specified LDIF file, if it exists.

`-b, --includeBranch` ***branchDN***
Specify the base DN for a branch or subtree of the data to be exported. This option can be used multiple times to specify multiple base DNs. If this option is provided, entries contained in the back end that are not at or below one of the provided base DNs are skipped.

`-B, --excludeBranch` ***branchDN***
Specify the base DN for a branch or subtree of the data to be omitted from the export. This option can be used multiple times to specify multiple base DNs. If this option is provided, any entries contained in the back end that are at or below one of the provided base DNs are skipped. Use of the `--excludeBranch` option takes precedence over the `--includeBranch` option. If an entry is at or below a DN contained in both the included and excluded lists, it is not included. This capability makes it possible to include data for only part of a branch. For example, you can include all entries below `dc=example,dc=com` except those below `ou=People,dc=example,dc=com`.

`-c, --compress`
Compress the LDIF data as it is written. The data is compressed using the GZIP format, which is the format used by the `--isCompressed` option of the `import-ldif` command.

`-d, --decrypt`
Decrypt the LDIF data as it is exported. The default value is not to decrypt.
If `-d` option is not used, then an encrypted attribute is exported encrypted and the presence of `{ENC}` header in the attribute value states that it is an encrypted value. The values that follows the `{ENC}` header is `base64` encoded format. Consider the following example, an LDIF entry with some encrypted attributes:

```
dn: uid=user87633,ou=Accounting,dc=example,dc=com objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: top uid: user87633 description: An employee of the company
userPassword:
{SSHA512}VYWWH4FxWtL7xez9Bz3n12Qvr9nnR1rwZa9tSjVk1EbZ8WgX0ay0ywPggQj2KnfABTdl9zYI/gjo+/
Z1ODbKVkKoharGfvfP
employeeNumber: 87633
pager: {ENC}AQ8F/ppNg0MArph6C+5upN9woi8A7+kPxvISoI+GqDUw
mobile: {ENC}AQ8F/ppNg0MArph6C+5upN+D113xXHRk5SPy2smCNyAn
mobile: {ENC}AQ8F/ppNg0MArph6C+5upN8RMqemKLxYPG09bkPUjBSk
```

Every string value following the `{ENC}` header is base64 encrypted format of the original value that is encrypted by the CryptoManager.
If `-d` option is used, then an encrypted attribute is exported in clear.

`-e, --excludeAttribute` *attribute*
Exclude the specified attribute name during the export. This option can be used multiple times to specify multiple attributes. If this option is provided, any attributes listed are omitted from the entries that are exported.

`-E, --excludeFilter` *filter*
Exclude the entries identified by the specified search filter during the export. This option can be used multiple times to specify multiple filters. If this option is provided, any entry in the back end that matches the filter is skipped. Use of the `--excludeFilter` option takes precedence over the `--includeFilter` option. If an entry matches filters in both the included and excluded lists, the entry is skipped.

`-i, --includeAttribute` *attribute*
Include the specified attribute name in the export. This option can be used multiple times to specify multiple attributes. If this option is provided, any attributes not listed are omitted from the entries that are exported.

`-g, --algorithm` *algorithm*
The specified algorithm used in the export. This option is optional and you can enter one of the following values:

- `diskOrder`: This option causes data to be read from an Oracle Berkeley DB Java Edition (JE) back end in the order that it is stored on the disk.

  Oracle recommends using the `diskOrder` option if the database does not fit entirely in the database cache. With this option, an export operation temporarily uses 20% of the database cache to run and then releases the memory. Thus, the database cache memory is decreased by 20% during the operation.

  > **Note:**
  >
  > This algorithm uses a feature called Disk Ordered from the JE back end and can cause an error if the server is running and you access it for modifications during the export operation. You can perform read operations.

- `entryIdOrder`: This option causes the data to be read from an Oracle Berkeley DB Java Edition (JE) back end in the order that it is logically stored on the disk.

  The `entryIdOrder` option provides better performance than the `diskOrder` option algorithm if the database fits entirely into the database cache.

  This option does not temporarily extract any memory from the database cache. Thus, you can use this option when the server is running and you want to access it for modifications during the export operation.

- `auto`: This option automatically selects `diskOrder` in an offline mode when the server is down or `entryIdOrder` in an online mode when the server is running.

`-I, --includeFilter` *filter*
Include the entries identified by the specified search filter in the export. This option can be used multiple times to specify multiple filters. If this option is provided, any entry in the back end that does not match the filter is skipped.

`-l, --ldifFile` *filename*
Export the data to the specified LDIF file. This is a required option.

For online exports, the root for relative paths is the *instance root*, rather than the current working directory. So, for example, a path of `exports/ldif.ldif` here refers to `instance-root/exports/ldif.ldif`.

`-n, --backendID` ***backendID***
Specify the back end ID of the data to be exported. The available back ends in the directory server can be determined using the `list-backends` command. This is a required option.

`-O, --excludeOperational`
Exclude operational attributes in the export.

`--wrapColumn` ***column***
Specify the column at which to wrap long lines when writing to the LDIF file. A value of `0` indicates that the data should not be wrapped.

Task Back End Connection Options

Running an online export requires access to the tasks back end. Access to the tasks back end is provided over SSL through the administration connector. These connection options are used when the export runs online.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***hostname***
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` ***path***
Use the client keystore certificate in the specified path.

`-N, --certNickname` ***nickname***
Use the specified certificate for client authentication.

`-o, --saslOption` ***name=value***
Use the specified options for SASL authentication.

`-p, --port` ***port***
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` ***path***
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

-U, --trustStorePasswordFile *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

-X, --trustAll
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

These options are used when you specify that the export should run as a scheduled task.

--completionNotify *emailAddress*
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

--dependency *taskId*
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

--errorNotify *emailAddress*
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

--failedDependencyAction *action*
Specify the action that this task will take if one of its dependent tasks fails. The value must be one of `PROCESS`, `CANCEL`, or `DISABLE`. If no value is specified, the default action is `CANCEL`.

--recurringTask *schedulePattern*
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

-t, --start *startTime*
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 schedules the task for immediate execution. When this option is specified, the operation is scheduled to start at the specified time after which the command exits immediately.

Command Input/Output Options

--noPropertiesFile
Indicates that a properties file is not used to obtain the default command-line options.

--propertiesFilePath *path*
Specify the path to the properties file that contains the default command-line options.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to run an export.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the directory server commands.

Performing an Offline Export

The following example exports the `userRoot` back end, starting at the base DN specified by the `-b` option. The command exports the data to an LDIF file specified by `-l`. The directory server must be stopped before performing an offline export.

```
$ stop-ds
$ export-ldif -b dc=example,dc=com -n userRoot -l /usr/tmp/export.ldif
[17/Oct/2008:12:24:33 +0200] category=JEB severity=NOTICE msgID=8847447
msg=Exported 102 entries and skipped 0 in 0 seconds (average rate 159.4/sec)
```

Performing an Online Export

An export is automatically run online if you specify any of the task back end connection options. Because an online export contacts the server over SSL, you must specify how to trust the SSL server certificate. This examples uses the `-X` option to trust all certificates.

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot \
  --ldifFile /usr/tmp/export.ldif
```

Scheduling an Export

You can schedule an export to run at some future date by using the `-t` or `--start` option to specify the start time. Like a regular online export, a scheduled export contacts the task back end of a running directory server and the relevant task back end connection options must be specified.

The following example schedules an export of the `userRoot` back end to start on December 24.

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot \
  --ldifFile /usr/tmp/export.ldif --start 20081224121500
Export task 2008101712361910 scheduled to start Dec 24, 2008 12:15:00 PM SAST
```

You can view a scheduled task by using the `manage-tasks` command. For more information, see Configuring Commands As Tasks.

Exit Codes

- **Offline mode**. An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.
- **Online mode**. If `-t` or `--start` is specified, an exit code of 0 indicates that the task was created successfully. A nonzero exit code indicates that an error occurred when the task was created. If `-t` or `--start` is not specified, the exit codes are the same as those specified for offline mode.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `export-ldif` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/export-ldif`
- Windows: *INSTANCE_DIR*`\OUD\bat\export-ldif.bat`

Related Commands

- import-ldif
- ldif-diff
- ldifmodify
- ldifsearch
- manage-tasks

A.1.3.6 import-ldif

The `import-ldif` command populates an Oracle Berkeley DB Java Edition (JE) back end with data that is read from an LDIF file.

Synopsis

`import-ldif` *options*

Description

The `import-ldif` command populates an Oracle Berkeley DB Java Edition (JE) back end with data that is read from an LDIF file, or with data generated based on a MakeLDIF template. In most cases, using `import-ldif` is significantly faster than adding entries by using `ldapmodify`. A complete import to an entire JE back end has better performance than a partial import to a branch of the JE back end.

The `import-ldif` command can run the import immediately or can schedule the import to run at a specified date and time. For more information, see Configuring Commands As Tasks.

You can run the `import-ldif` command in online or offline mode.

- **Online mode**. In online mode, `import-ldif` contacts a running directory server instance over SSL, through the administration connector, and registers an import task. The command runs in online mode automatically if you specify any of the task back end connection options. For more information about the administration connector, see Managing Administration Traffic to the Server.
- **Offline mode**. In offline mode, `import-ldif` accesses the database directly rather than through a directory server instance. To perform an offline import, the directory server must be stopped.

Options

The `import-ldif` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--includeBranch` *baseDN*).

`-a, --append`
Append the imported data to the data that already exists in the back end, rather than clearing the back end before starting the import.

`-A, --templateFile` *filename*
Specify the path to a MakeLDIF template to generate the import data.

`-b, --includeBranch` *branchDN*
Specify the base DN for a branch or subtree of the data that should be included in the import. This option can be used multiple times to specify multiple base DNs. If this option is provided, entries contained in the import source that are not at or below one of the provided base DNs are skipped. Any existing entries above the provided base DNs are preserved.

`-B, --excludeBranch` *branchDN*
Specify the base DN branch or subtree that should be omitted from the import. This option can be used multiple times to specify multiple base DNs. If this option is provided, entries contained in the import source that are at or below one of the base DNs are skipped. Use of the `--excludeBranch` option takes precedence over the `--includeBranch` option. If an entry is at or below a DN contained in both the included and excluded lists, it is omitted from the import. This capability makes it possible to include data for only a part of a branch (for example, all entries below `dc=example,dc=com` except those below `ou=People,dc=example,dc=com`).

`-c, --isCompressed`
Specify that the LDIF import file is compressed. The file should be compressed using the GZIP format, which is the format used by the `--compressLDIF` option of the `export-ldif` command.

`--countRejects`
Return the number of rejected entries during import. If the number of rejected entries is between 0 and 255, that number is returned. If the number of rejected entries is greater than 255, the command returns the value `255`. For example, if you run `import-ldif` with the `--countRejects` option and get 16 rejected entries, the command returns the value `16`. If you run `import-ldif` and get 300 rejected entries, the command returns the value `255`.

> **Note:**
>
> This option is not supported for online imports.

`-e, --excludeAttribute` *attribute*
Specify the name of an attribute that should be excluded from the import. This option can be used multiple times to specify multiple attributes.

`-E, --excludeFilter` *filter*
Specify the search filter to identify entries that should be excluded from the import. This option can be used multiple times to specify multiple filters. If this option is provided, any entry in the import source that matches the filter is skipped. The `--excludeFilter` option takes precedence over the `--includeFilter` option. If an entry matches filters in both the include and exclude filters, the entry is skipped during import.

`-F, --clearBackend`
Confirm deletion of all existing entries for all base DNs in the specified back end when importing without the `--append` option. This only applies when importing a multiple base DN back end specified by the back end ID. This option is implied for back ends with only one base DN.

`-i, --includeAttribute` *attribute*
Specify the attributes that should be included in the import. This option can be used multiple times to specify multiple attributes. If this option is used, attributes not listed in this set are omitted from the entries that are imported.

`-I, --includeFilter` *filter*
Specify the search filter to identify entries that should be included in the import. This option can be used multiple times to specify multiple filters. If this option is provided, any entry in the import source that does not match the results of the filter is skipped.

`-l, --ldifFile` *filename*
Read the LDIF file located at the specified path. Do not use this option with `--templateFile`. For online imports, the root for relative paths is the *instance root*, rather than the current working directory. So, for example, a path of `imports/ldif.ldif` here refers to `instance-root/imports/ldif.ldif`.

`-n, --backendID` *backendID*
Specify the ID of the back end into which the data should be imported. To display the available back ends in the server, use the `list-backends` command.

`-O, --overwrite`
Overwrite the specified skip file or reject file, if it already exists. If this option is not provided, any skipped or rejected entries are appended to their corresponding files rather than overwriting them. This option is only applicable if the `--rejectFile` or `--skipFile` options are provided.

`-r, --replaceExisting`
Replace existing data with the content from the import. If this option is not provided, existing entries are not overwritten. This is only applicable if the `--append` option has also been provided.

`-R, --rejectFile` *filename*
Use the specified file to hold any rejected entries during the import. Rejected entries occur if entries are not compliant with the default schema. A comment is included before the entry indicating the reason that it was rejected. If this option is not provided, no reject file is written.

`-s, --randomSeed` *seed*
Use the specified seed number for the random number generator when generating entries from a MakeLDIF template. Seeding the random number generator with a particular value can help to ensure that the same template and random seed always generate exactly the same data.

`--skipDNValidation`
Perform limited parental DN validation during a later part of the LDIF import. If this option is specified, no duplicate DN checking is done. Do not use this option if you are not certain that your LDIF import file is correct.

`--skipFile filename`
Use the specified file to identify entries that were skipped during the import. Skipped entries occur if entries cannot be placed under any specified base DN during an import or if the `--excludeBranch`, `--excludeAttribute`, or `--excludeFilter` option is used.

`-S, --skipSchemaValidation`
Do not perform any schema validation on the entries as they are imported. This option can provide improved import performance, but should only be used if you are certain that the import data is valid.

`--threadCount` *count*
Specify the number of threads that are used to read the LDIF file. If this option is not specified, a default of two threads per CPU is used.
You can use this option to increase the number of threads if you are importing particularly large LDIF files, but you should not use the option unless you are certain of the resulting impact on performance.

`--tmpDirectory` *directory*
Use the specified directory for index scratch files created during the import. If no directory is specified, the default *INSTANCE_DIR*`/OUD/import-tmp` is used.

Task Back End Connection Options

Running an online import requires access to the tasks back end. Access to the tasks back end is provided over SSL through the administration connector. These connection options are used when the import runs online.

`-D, --bindDN` *bindDN*
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` *hostname*
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` *filename*
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-N, --certNickname` *nickname*
Use the specified certificate for client authentication.

`-o, --saslOption` *name=value*
Use the specified options for SASL authentication.

`-p, --port` *port*
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `6664` is used.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

-X, --trustAll
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

These options are used when you specify that the import should run as a scheduled task.

--completionNotify *emailAddress*
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

--dependency *taskId*
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

--errorNotify *emailAddress*
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

--failedDependencyAction *action*
Specify the action that this task will take if one of its dependent tasks fails. The value must be one of `PROCESS`, `CANCEL`, or `DISABLE`. If no value is specified, the default action is `CANCEL`.

--recurringTask *schedulePattern*
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

-t, --start *startTime*
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 schedules the task for immediate execution. When this option is specified, the operation is scheduled to start at the specified time after which the command exits immediately.

Command Input/Output Options

--noPropertiesFile
Indicates that a properties file is not used to obtain the default command-line options.

--propertiesFilePath *path*
Specify the path to the properties file that contains the default command-line options.

-Q, --quiet
Run in quiet mode. Using quiet mode, no output is generated unless a significant error occurs during the import process.

-d, --debug
Use debug mode (verbose). Using debug mode, all advanced or debug messages are output.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to run an import.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the directory server commands.

Running an Offline Import

This example imports an LDIF file to the `userRoot` back end. The LDIF file path supports both absolute and relative paths on all platforms. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif
```

Importing Part of an LDIF File Offline

This example imports part of an LDIF file to the `userRoot` back end. The import includes the base DN `dc=example,dc=com` but excludes the branch `ou=people`. Existing entries are replaced (`-r`) and information about any rejected entries are written to `/usr/tmp/rejects.ldif`. The LDIF file path supports both absolute and relative paths on all platforms. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -B "ou=people,dc=example,dc=com" \
  -l /usr/tmp/Example.ldif -n userRoot -r -R /usr/tmp/rejects.ldif
```

Importing Data From a MakeLDIF Template

This example imports sample data from a MakeLDIF template to the `userRoot` back end. The random seed (`-s`) determines the randomness of the data. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -n userRoot -A example.template -s 0
```

Importing User Attributes Only

This example imports an LDIF file to the `userRoot` back end. Only user attributes are imported, specified by `-i "*"`. The LDIF file path supports both absolute and relative paths on all platforms. On some systems, you might be required to enclose the asterisk in quotation marks (`"*"`) or to escape the asterisk using a character appropriate to your shell. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif -i "*"
```

Importing User Attributes and Excluding an Attribute

This example imports an LDIF file to the `userRoot` back end. All user attributes are imported, specified by `-i "*"`, but the `roomnumber` attribute is excluded. The LDIF file path supports both absolute and relative paths on all platforms. On some systems, you might be required to enclose the asterisk in quotation marks (`"*"`) or to escape the asterisk using a character appropriate to your shell. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif \
  -i "*" -e "roomnumber"
```

Importing Operational Attributes Only

This example imports an LDIF file to the `userRoot` back end. Only operational attributes are imported, specified by `-i "+"`. The LDIF file path supports both absolute and relative paths on all platforms. On some systems, you might be required to enclose the plus sign in quotation marks (`"+"`) or to escape the plus sign using a character appropriate to your shell. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif -i "+"
```

Importing Selected User and Operational Attributes

This example imports an LDIF file to the `userRoot` back end. Only the `uid`, `cn`, `sn`, `dc`, and `creatorsname` attributes are imported. The LDIF file path supports both absolute and relative paths on all platforms. The directory server must be stopped before running an offline import.

```
$ stop-ds
$ import-ldif -b dc=example,dc=com -n userRoot -l /var/tmp/Example.ldif \
  -i "uid" -i "cn" -i "sn" -i "dc" -i "creatorsname"
```

Running an Online Import

An import is automatically run online if you specify any of the task back end connection options. Because an online import contacts the server over SSL, you must specify how to trust the SSL server certificate. This examples uses the `-X` option to trust all certificates.

```
$ import-ldif -h localhost -p 6664 -D "cn=Directory Manager" -j /path/pwd-file \
  -X -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif
```

Scheduling an Import

You can schedule an import to run at some future date by using the `-t` or `--start` option to specify the start time. Like a regular online import, a scheduled import contacts the task back end of a running directory server and the relevant task back end connection options must be specified.

The following example schedules an import to the `userRoot` back end to start on December 24.

```
$ import-ldif -h localhost -p 6664 -D "cn=Directory Manager" -j /path/pwd-file \
  -X -b dc=example,dc=com -n userRoot -l /usr/tmp/Example.ldif \
  --start 20081224121500
Import task 2008101712361910 scheduled to start Dec 24, 2008 12:15:00 PM SAST
```

You can view a scheduled task by using the `manage-tasks` command. For more information, see Configuring Commands As Tasks.

Exit Codes

- **Offline mode**. An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.
- **Online mode**. If `-t` or `--start` is specified, an exit code of 0 indicates that the task was created successfully. A nonzero exit code indicates that an error occurred when the task was created. If `-t` or `--start` is not specified, the exit codes are the same as those specified for offline mode.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `export-ldif` command. The properties file is convenient when working in

different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/import-ldif`
- Windows: *INSTANCE_DIR*`\OUD\bat\import-ldif.bat`

Related Commands

- export-ldif
- ldif-diff
- ldifmodify
- ldifsearch
- manage-tasks

A.1.3.7 ldif-diff

The `ldif-diff` command identifies the differences between two LDIF files.

Synopsis

`ldif-diff` *options*

Description

The `ldif-diff` command can be used to identify the differences between two LDIF files. The resulting output can be displayed on the terminal or saved to an output file. The resulting output contains all of the information necessary for someone to reverse any changes if necessary. For modify operations, only sets of `add` and `delete` change types are used, not the `replace` change type. For delete operations, the contents of the entry that has been removed are included in the changes displayed in the form of comments.

This command was designed to work on small data sets. It is only suitable in cases in which both the source and target data sets can fit entirely in memory at the same time. It is not intended for use on large data sets that cannot fit in available memory.

> **Note:**
>
> The `ldif-diff` command is not intended for large files. Running the `ldif-diff` command on LDIF files over a certain size (around 600 Kbytes on Windows systems, larger on UNIX systems) might result in a memory error similar to the following:
>
> ```
> Exception in thread "main" java.lang.OutOfMemoryError: Java heap space.
> ```

Options

The `ldif-diff` command accepts an option in either its short form (for example, `-o` *outputFile*) or its long form equivalent (for example, `--outputLDIF` *outputFile*).

`-a, --ignoreAttrs` *file*
Specify a file containing a list of attributes to ignore when computing the difference

`--checkSchema`
Consider the syntax of the attributes as defined in the schema to make the value comparison. The specified LDIF files must be conform to the server schema.

`-e, --ignoreEntries` ***file***
Specify a file containing a list of entries (DNs) to ignore when computing the difference

`-o, --outputLDIF` ***outputLDIF***
Specify the path to the output file to record the changes between the source and target LDIF data. If this is not provided, then the change information will be written to standard output.

`-O, --overwriteExisting`
Overwrite the output file specified with the `--outputLDIF` option. This option indicates that if the specified output file already exists that the file should be overwritten rather than appending to it. The option is only applicable if `--outputLDIF` is used.

`-s, --sourceLDIF` ***sourceLDIF***
Specify the path to the source LDIF file, which contains the original data with no changes applied. This option is required.

`-S, --singleValueChanges`
Run in *Single Value Change* mode, in which each modify operation is broken into a separate modification per attribute value. For example, if a single modification adds five values to an attribute, the changes appear in the output as five separate modifications, each adding one attribute.

`-t, --targetLDIF` ***targetLDIF***
Specify the path to the target LDIF file that contains the differences from the source LDIF. This option is required.

`-?, -H, --help`
Display command usage information and exit without attempting to perform any additional processing.

`-V, --version`
Display the directory server version information and exit rather than attempting to run this command.

Examples

The following examples show how to use the `ldif-diff` command.

Comparing Two LDIF files and Sending the Differences to Standard Output

The following command compares a source file (`-s`) with a target file (`-t`) and outputs the differences. For Windows platforms, specify the paths for the source file (for example, `-s \temp\quentin.ldif`) and the target file (for example, `-t \temp\quentin.ldif`):

```
$ ldif-diff -s /usr/local/quentin.ldif -t /usr/local/quentinr.ldif

dn: uid=qcubbins,ou=People,dc=example,dc=com
changetype: delete
# objectClass: person
# objectClass: organizationalPerson
# objectClass: top
# objectClass: inetOrgPerson
# cn: Quentin Cubbins
# sn: Cubbins
# uid: qcubbins
```

```
# userPassword: qcubbins
# givenName: Quentin
# description: This is Quentin's description.
# mail: qcubbins@example.com

dn: uid=qrcubbins,ou=People,dc=example,dc=com
changetype: add
objectClass: person
objectClass: organizationalPerson
objectClass: top
objectClass: inetOrgPerson
cn: Quentin R Cubbins
sn: Cubbins
uid: qrcubbins
userPassword: qrcubbins
givenName: Quentin
description: This is Quentin R's description.
mail: qrcubbins@example.com
```

Comparing Two LDIF files and Sending the Differences to a File

The following command compares a source file (`-s`) with a target file (`-t`) and sends the output to a file (`-o`). For Windows platforms, specify the paths for the source file (for example, `-s \temp\quentin.ldif`) and the target file (for example, `-t \temp\quentin.ldif`):

```
$ ldif-diff -s /usr/local/quentin.ldif -t /usr/local/quentinr.ldif \
  -o output.ldif
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 or greater indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldif-diff`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldif-diff.bat`

Related Commands

- ldifsearch
- ldifmodify
- make-ldif

A.1.3.8 ldifmodify

The `ldifmodify` command makes changes to the contents of an LDIF file.

Synopsis

`ldifmodify` *options*

Description

The `ldifmodify` command can be used to make changes to the contents of an LDIF file. Although similar to the `ldapmodify` command, the `ldifmodify` command does not connect to the directory server but rather operates locally on the LDIF file. The command also does not accept change information on standard input. It must read all changes from a file.

To make it possible to operate on very large LDIF files with limited amounts of memory, the following limitations will be enforced on the types of changes that can be made:

- **No modify DNs**. Modify DN operations are not supported. Only add, delete, and modify operations will be allowed.
- **No concurrent modify or delete operations**. It is not possible to modify or delete an entry that is to be added during processing.

Options

All options (with the exception of `--help` and `--version`) are required. The `ldifmodify` command accepts an option in either its short form (for example, `-m` *changeFile*) or its long form equivalent (for example, `--changesLDIF` *changeFile*).

`-m, --changesLDIF` ***changeFile***
Specify the path to the file containing the changes to apply. The contents of this file must be in LDIF change format.

`-s, --sourceLDIF` ***sourceFile***
Specify the path to the source LDIF file, which contains the data to be updated.

`-t, --targetLDIF` ***targetFile***
Specify the path to the target LDIF file, which will consist of the data from the source LDIF with all of the specified changes applied.

`-?, -H, --help`
Display command usage information and exit without attempting to perform any additional processing.

`-V, --version`
Display the directory server version information and exit rather than attempting to run this command.

Examples

The following examples show how to use the `ldifmodify` command.

Modifying an LDIF File

Suppose that the source file is as follows:

```
dn: uid=qcubbins,ou=People,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
uid: qcubbins
givenName: Quentin
sn: Cubbins
cn: Quentin Cubbins
mail: qcubbins@example.com
userPassword: qcubbins
description: This is Quentin's description.
```

And suppose that the update (change) file is as follows:

```
## Add new telephone number for Quentin Cubbins
dn: uid=qcubbins,ou=People,dc=example,dc=com
changetype: modify
```

```
add: telephoneNumber
telephoneNumber: 512-401-1241
```

The following command updates a source file (`-s`) with changes listed in a modify file (`-m`) and outputs to a target file (`-t`). For Windows platforms, use the file paths for the modify file (for example, `-m \temp\update.ldif`), the source file (for example, `-s \temp\quentin.ldif`), and the target file (for example, `-s \temp\quentin_updated.ldif`):

```
$ ldifmodify -m /usr/local/update.ldif -s /usr/local/quentin.ldif \
-t /usr/local/quentin_updated.ldif
```

The updated file is as follows:

```
dn: uid=qcubbins,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
objectClass: person
objectClass: top
objectClass: organizationalPerson
sn: Cubbins
userPassword: qcubbins
description: This is Quentin's description.
cn: Quentin Cubbins
telephoneNumber: 512-401-1241
givenName: Quentin
uid: qcubbins
mail: qcubbins@example.com
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 or greater indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldifmodify`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldifmodify.bat`

Related Commands

- ldifsearch
- ldif-diff
- make-ldif

A.1.3.9 ldifsearch

The `ldifsearch` command performs searches in an LDIF file.

Synopsis

`ldifsearch` [*options*]

Description

The `ldifsearch` command can be used to perform searches in an LDIF file. Although similar to the `ldapsearch` command, the `ldifsearch` command does not perform any LDAP communication with the directory server but rather operates locally on the LDIF file.

Options

The `ldifsearch` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--baseDN` *baseDN*).

`-b, -baseDN` ***baseDN***
Specify the base DN to use for the search operation. Multiple base DNs can be provided by using this option multiple times. If multiple values are provided, then an entry will be examined if it is within the scope of any of the search bases. If no search base is provided, then any entry contained in the LDIF files will be considered in the scope of the search.

`-f, --filterFile` ***filterFile***
Specify the path to a file containing one or more filters to use when processing the search operation. If there are to be multiple filters, then the file should be structured with one filter per line. If this option is used, then any trailing options will be treated as separate attributes. Otherwise, the first trailing option must be the search filter.

`-l, -ldifFile` ***ldifFile***
Specify the path to the LDIF file containing the data to be searched. Multiple LDIF files can be specified by providing this option multiple times. This option is required.

`-o, -outputFile` ***outputFile***
Specify the path to the output file that contains the entries matching the provided search criteria. If this option is not provided, the matching entries will be written to standard output.

`-O, --overwriteExisting`
Overwrite the output file specified with the `--outputFile` option. This option indicates that if the specified output file already exists that the file should be overwritten rather than appending the data to existing data. This is only applicable if the `--outputFile` option is used.

`-s, -searchScope` ***searchScope***
Specify the scope of the search operation. Its value must be one of the following:

- `base` Examine only the entry specified by the `--baseDN` option.
- `one` Examine only the entry specified by the `--baseDN` option and its immediate children.
- `sub` or `subordinate` Examine the entry specified by the `--baseDN` option and its subtree.

Default value `sub` if the option is not specified.

`-t, --timeLimit` ***numSeconds***
Indicate the maximum length of time in seconds that should be spent performing the searches. After this length of time has elapsed, the search ends.

`-z, --sizeLimit` ***sizeLimit***
Set the maximum number of matching entries that the directory server should return to the client. If this is not provided, then there will be no maximum requested by the client.

> **Note:**
>
> The directory server can enforce a lower size limit than the one requested by the client.

-T, --dontWrap
Do not wrap long lines when displaying matching entries. If this option is not provided, long lines will be wrapped (in a manner compatible with the LDIF specification) to fit on an 80-column terminal.

-?, -H, --help
Display command usage information and exit without attempting to perform any additional processing.

-V, --version
Display the version information for the directory server.

Examples

The following examples show how to use the `ldifsearch` command.

Searching an LDIF File

The following command specifies the base DN (`-b`) and searches an LDIF file (`-l`) for an entry and returns its result to the screen if any entries match the search filter `cn=Sam Carter`. For Windows platforms, use the path where the LDIF file resides (for example, `-l \temp\Example.ldif`.

```
$ ldifsearch -b dc=example,dc=com -l /usr/local/Example.ldif "(cn=Sam Carter)"

dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
objectClass: person
objectClass: top
objectClass: organizationalPerson
ou: Accounting
ou: People
sn: Carter
facsimiletelephonenumber: +1 408 555 9751
roomnumber: 4600
userpassword: sprain
l: Sunnyvale
cn: Sam Carter
telephonenumber: +1 408 555 4798
uid: scarter
givenname: Sam
mail: scarter@example.com
```

Searching an LDIF File by Using a Filter File

Suppose that the file, `filter.ldif`, which contains the following search filter:

```
(&(ou=Accounting)(l=Cupertino))
```

The following command searches the LDIF file for entries that match the filter in the search filter file and outputs the results in an output file. The command specifies the base DN (`-b`) and searches the LDIF file (`-l`) using the search filter file (`-f`) and outputs the results in a file (`-o`). For Windows platforms, use the file paths for the LDIF file (for example, `-l \temp\Example.ldif`), the filter file (for example, `-f \temp\filter.ldif`), and the output file (for example, `-o \temp\results.ldif`):

```
$ ldifsearch -b dc=example,dc=com -l /usr/local/Example.ldif -f /usr/local/filter.ldif \
-o /home/local/results.ldif
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 or greater indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldifsearch`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldifsearch.bat`

Related Commands

- ldifmodify
- ldif-diff

A.1.3.10 list-backends

The `list-backends` command displays information about the available back ends.

Synopsis

`list-backends` [*options*]

Description

The `list-backends` command can be used to obtain information about the back ends defined in a directory server instance. Back ends are responsible for providing access to the server database.

The `list-backends` command has three modes of operation:

- **No options.** When invoked with no options, display the back-end IDs for all back ends configured in the server, along with the base DNs for those back ends.
- **With backend ID.** When used with the `--backendID`, list all of the base DNs for the back end with the specified back-end ID.
- **With baseDN.** When used with the `--baseDN` option, list the back-end ID of the back end that should be used to hold the entry with the given DN and also indicate whether that DN is one of the configured base DNs for that back end.

Options

The following are available for use but are not required. The `list-backends` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--baseDN` *baseDN*).

Command Options

`-b, --baseDN` *baseDN* Specify the base DN from which the `list-backends` command should list the back-end ID. The option also indicates whether the specified DN is a baseDN for that back end.

`-n, --backendID` *backendID* Specify the back-end ID from which the command should display the associated base DN. This option can be used multiple times to display the base DNs for multiple back ends.

General Options

`-?, -H, --help` Display the command usage information and exit immediately without taking any other action.

`-V, --version` Display the directory server version information and exit rather than attempting to run this command.

Examples

The following examples show how to use the `list-backends` command.

Listing the Current Back Ends

The following command lists the current back ends on the directory server:

```
$ list-backends

Backend ID  Base DN
----------  -----------------
backup      cn=backups
config      cn=config
monitor     cn=monitor
schema      cn=schema
tasks       cn=tasks
userRoot    dc=example,dc=com
```

Listing the Back-end ID

The following command lists the back-end ID on the directory server:

```
$ list-backends --backendID monitor

Backend ID  Base DN
----------  ----------
monitor     cn=monitor
```

Listing the Base DN

The following command lists the base DN on the directory server:

```
$ list-backends --baseDN cn=backups

The provided DN 'cn=backups' is a base DN for the back end 'backup'
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/list-backends`
- Windows: *INSTANCE_DIR*`\OUD\bat\list-backends.bat`

A.1.3.11 make-ldif

The `make-ldif` command generates LDIF data based on a template file.

Synopsis

`make-ldif` [*options*]

Description

The `make-ldif` command can be used to generate LDIF data based on a template file. The command allows you to construct any amount of realistic sample data that is suitable for use in applications, such as performance and scalability testing, or to attempt to reproduce a problem observed in a production environment.

Options

The `make-ldif` command accepts an option in either its short form (for example, `-o` *ldifFile*) or its long form equivalent (for example, `--ldifFile` *ldifFile*).

`-o, --ldifFile` ***ldifFile***
Specify the path to the LDIF file to which the generated data should be written. This is a required option.

`-s, --randomSeed` ***seed***
Specify the integer value that should be used to seed the random number generator. If a random seed is provided, then generating data based on the same template file with the same seed will always generate exactly the same LDIF output. If no seed is provided, then the same template file will likely generate different LDIF output each time it is used.

`-t, --templateFile` ***templateFile***
Specify the path to the template file that describes the data to be generated. This is a required option. You must specify an absolute path to the template file.

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to run the command.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the `make-ldif` command.

Creating a Sample LDIF File

The following command creates an LDIF file using the template (`-t`), writes to an output file (`-o`), and specifies the random seed (`-s`). For Windows platforms, enter the file paths to your output LDIF file (for example, `-o path\to\Example.ldif`) and to your template file (for example, `-t` *INSTANCE_DIR*`\OUD\config\MakeLDIF\example.template`).

The `example.template` file is located in the *INSTANCE_DIR*`/OUD/config/MakeLDIF` directory.

```
$ make-ldif -o /path/to/sample.ldif -s 0 \
-t INSTANCE_DIR/OUD/config/MakeLDIF/example.template

Processed 1000 entries
Processed 2000 entries
Processed 3000 entries
Processed 4000 entries
Processed 5000 entries
```

```
Processed 6000 entries
Processed 7000 entries
Processed 8000 entries
Processed 9000 entries
Processed 10000 entries
LDIF processing complete.  10003 entries written
```

Creating a Large Sample LDIF File

The `example.template` file (located in the installation directory under *INSTANCE_DIR*`/OUD/config/MakeLDIF`) contains a variable that sets the number of entries generated by the `make-ldif` command. You can change the number to create a very large sample LDIF file for your tests.

Open the `example.template` file, and change the `numusers` variable. By default, the variable is set to `10001`. In this example, set the variable to `1000001`:

```
define suffix=dc=example,dc=com
define maildomain=example.com
define numusers=1000001
...
```

Rerun the `make-ldif` command:

```
$ make-ldif -o /path/to/sample.ldif -s 0 \
-t INSTANCE_DIR/OUD/config/MakeLDIF/example.template
...
Processed 999000 entries
Processed 1000000 entries
LDIF processing complete.  1000003 entries written
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

Locations

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/make-ldif`
- Windows: *INSTANCE_DIR*`\OUD\bat\make-ldif.bat`

Related Commands

- ldifsearch
- ldifmodify
- ldif-diff

A.1.3.12 manage-account

The `manage-account` command manages user account information, primarily related to password policy state details.

Synopsis

`manage-account` *subcommands options*

Description

The `manage-account` command manages user account information, primarily related to password policy state details. The command interacts with the Password Policy State extended operation, which returns account, login, and password information for a user. Although the Password Policy State extended operation allows multiple operations per use, the `manage-account` command can run only one operation at a time. Users must have the `password-reset` privilege to use the Password Policy State extended operation.

> **Note:**
>
> All time values are returned in generalized time format. All duration values are returned in seconds.

The `manage-account` command connects to the server over SSL through the administration connector (described in Managing Administration Traffic to the Server).

Subcommands

`clear-account-is-disabled`
Clear the disabled state for the user account. This will have the effect of enabling the account if it is disabled.

`get-account-expiration-time`
Return the account expiration time.

`get-account-is-disabled`
Return the disabled state for the user account.

`get-all`
Return all Password Policy State information for the user account.

`get-authentication-failure-times`
Return the authentication failure times for the user account.

`get-grace-login-use-times`
Return the grace login use times for the user account.

`get-last-login-time`
Return the last login time for the user.

`get-password-changed-by-required-time`
Return the password changed by the required time for the user.

`get-password-changed-time`
Return the time the password was last changed.

`get-password-expiration-warned-time`
Return the time the user was first warned about an upcoming password expiration.

`get-password-history`
Return the password history for the user account.

`get-password-is-reset`
Return the password reset state for the user, which indicates whether the user will be forced to change his password on the next login.

`get-password-policy-dn`
Return the DN of the password policy for a given user.

`get-remaining-authentication-failure-count`
Return the number of remaining authentication failures for the user before the user's account is locked.

`get-remaining-grace-login-count`
Return the number of remaining grace logins for the user.

`get-seconds-until-account-expiration`
Return the length of time before the account expires.

`get-seconds-until-authentication-failure-unlock`
Return the length of time before the user's account is automatically unlocked.

`get-seconds-until-idle-lockout`
Return the length of time before the account is idle-locked.

`get-seconds-until-password-expiration`
Return the length of time before the password expires.

`get-seconds-until-password-expiration-warning`
Return the length of time before the user is first warned about an upcoming password expiration.

`get-seconds-until-password-reset-lockout`
Return the length of time before the password reset lockout occurs.

`get-seconds-until-required-change-time`
Return the length of time before the user is required to change his password due to the required change time.

`set-account-is-disabled`
Disable the account. Required suboption:
`--operationValue` *true/false*. If set to `TRUE`, disable the user. If set to `FALSE`, enable the user.

Options

The `manage-account` command accepts an option in either its short form (for example, `-b` *targetDN*) or its long form equivalent (for example, `--targetDN` *targetDN*).

`-b, --targetDN` ***targetDN***
Specify the DN of the user entry for which to get and set password policy state information.

LDAP Connection Options

The `manage-account` command contacts the directory server over SSL through the administration connector. These connection options are used to contact the directory server.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` *hostname*
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` *filename*
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` *path*
Use the client keystore certificate in the specified path.

`-N, --certNickname` *nickname*
Use the specified certificate for client authentication.

`-o, --saslOption` *name=value*
Use the specified options for SASL authentication.

`-p, --port` *port*
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to run the command.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the directory server commands.

Viewing All Password Policy State Information for a User

The following command returns the password policy state information for a user:

```
$ manage-account get-all -h localhost -p 4444 -D "cn=Directory Manager" \
  -j /path/pwd-file -X -b "uid=scarter,ou=People,dc=example,dc=com" \
```

```
Password Policy DN:  cn=Default Password Policy,cn=Password Policies,cn=config
Account Is Disabled:  false
Account Expiration Time:
Seconds Until Account Expiration:
Password Changed Time:  19700101000000.000Z
Password Expiration Warned Time:
Seconds Until Password Expiration:
Seconds Until Password Expiration Warning:
Authentication Failure Times:
Seconds Until Authentication Failure Unlock:
Remaining Authentication Failure Count:
Last Login Time:
Seconds Until Idle Account Lockout:
Password Is Reset:  false
Seconds Until Password Reset Lockout:
Grace Login Use Times:
Remaining Grace Login Count:  0
Password Changed by Required Time:
Seconds Until Required Change Time:
```

Disabling a User Account

The following command disables a user's account `uid=scarter`:

```
$ manage-account set-account-is-disabled --operationValue true \
  -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
  -b "uid=scarter,ou=People,dc=example,dc=com"

Account Is Disabled:  true
```

Enabling a User Account

The following command re-enables a user's disabled account:

```
$ manage-account clear-account-is-disabled \
  -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
  -b "uid=scarter,ou=People,dc=example,dc=com"

Account Is Disabled:  false
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/manage-account`
- Windows: *INSTANCE_DIR*`\OUD\bat\manage-account.bat`

Related Commands

- verify-index

A.1.3.13 rebuild-index

The `rebuild-index` command rebuilds a directory server index.

Synopsis

`rebuild-index` *options*

Description

The `rebuild-index` command is used to rebuild directory server indexes. Indexes are files that contain lists of values, where each value is associated with a list of entry identifiers to suffixes in the directory server database. When the directory server processes a search request, it searches the database using the list of entry identifiers in the indexes, thus speeding up the search. If indexes did not exist, the directory server would have to look up each entry in the database, which dramatically degrades performance.

The `rebuild-index` command is useful in the following cases:

- When the `index-entry-limit` property of an index changes
- When a new index is created

The `rebuild-index` command can be run with the server online. However, the back-end database is unavailable while `rebuild-index` is running.

> **Note:**
>
> Online option is useful when there are multiple back-ends.
>
> With online option, `rebuild-index` can be executed separately for different back-ends without bringing down all the back-ends.

The `rebuild-index` command usually runs faster with the server offline, especially if the `--rebuildAll` option is specified.

> **Note:**
>
> As time progresses, the list of entry identifiers becomes unordered. As this happens, the performance of the `rebuild-index` command gradually decreases.
>
> If you can avoid reindexing large databases, you should do so. Otherwise, if the performance of the `rebuild-index` command is severely compromised, reimport the database, to start with a fresh, ordered list of entry identifiers.

Options

The `rebuild-index` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--baseDN` *baseDN*).

Command Options

`-b, --baseDN` *baseDN*
Specify the base DN of a back end that supports indexing. The rebuild operation is performed on indexes within the scope of the given base DN.

`-i, --index` *index*
Specify the name of the indexes to rebuild. For an attribute index, this is simply an attribute name. At least one index must be specified for rebuild.

`--rebuildAll`
Rebuild all indexes that are contained in the back end that is specified by the base DN. This option not only re-indexes all attribute indexes but also the `dn2id` system index, any extensible and VLV indexes, and the `dn2uri` index. The `rebuildAll` option cannot be used with the `-i` option.

`--tmpDirectory`
Specify the location of a temporary work directory for scratch index files. The default temporary work directory is *INSTANCE_DIR*`/OUD/import-tmp`.

Task Back End Connection Options

Rebuilding an index online requires access to the tasks back end. Access to the tasks back end is provided over SSL through the administration connector. These connection options are used when the rebuild runs online.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***hostname***
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` ***path***
Use the client keystore certificate in the specified path.

`-N, --certNickname` ***nickname***
Use the specified certificate for client authentication.

`-o, --saslOption` ***name=value***
Use the specified options for SASL authentication.

`-p, --port` ***port***
Contact the directory server at the specified administration port. If this option is not provided, the default administration port of `4444` is used.

`-P, --trustStorePath` ***path***
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

-X, --trustAll
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

These options are used when you specify that the index should be rebuilt as a scheduled task.

--completionNotify *emailAddress*
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

--dependency *taskId*
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

--errorNotify *emailAddress*
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

--failedDependencyAction *action*
Specify the action that this task will take if one of its dependent tasks fails. The value must be one of `PROCESS`, `CANCEL`, or `DISABLE`. If no value is specified, the default action is `CANCEL`.

--recurringTask schedulePattern
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

-t, --start *startTime*
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 schedules the task for immediate execution. When this option is specified, the operation is scheduled to start at the specified time after which the command exits immediately.

Utility Input/Output Options

--propertiesFilePath *propertiesFilePath*
Path to the file containing default property values used for command line

--noPropertiesFile
No properties file will be used to get default command line argument values.

-v, --verbose
Use verbose mode.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `rebuild-index` command.

Rebuilding an Index

First, display a list of indexes by using the `dsconfig` command as follows:

```
$ dsconfig -h localhost -p 4444 -D "cn=Directory Manager" -j /path/pwd-file -X \
  -n list-local-db-indexes --element-name userRoot

Local DB Index    : Type    : index-type
------------------:---------:--------------------
aci               : generic : presence
cn                : generic : equality, substring
displayName       : generic : equality, substring
ds-sync-conflict  : generic : equality
ds-sync-hist      : generic : ordering
entryUUID         : generic : equality
givenName         : generic : equality, substring
mail              : generic : equality, substring
member            : generic : equality
objectClass       : generic : equality
orclMTTenantGuid  : generic : equality
orclMTTenantUName : generic : equality, substring
orclMTUid         : generic : equality
sn                : generic : equality, substring
telephoneNumber   : generic : equality, substring
uid               : generic : equality
uniqueMember      : generic : equality
```

The following command rebuilds indexes (`-i`) with a base DN (`-b`).

Because this command runs offline, the directory server must be stopped before you run it.

```
$ rebuild-index -b dc=example,dc=com -i uid -i mail
[15/Dec/2011:15:28:01 +0100] category=JEB severity=NOTICE msgID=8847497
  msg=Rebuild of index(es) uid started with 202 total entries to process
...
[15/Dec/2011:15:28:02 +0100] category=JEB severity=NOTICE msgID=8847493
  msg=Rebuild complete. Processed 202 entries in 1 seconds (average rate 135.2/sec)
```

Rebuilding All Indexes

This example uses the `--rebuildAll` option to rebuild all indexes.

```
$ rebuild-index -b "dc=example,dc=com" --rebuildAll
```

Rebuilding Extensible Indexes

You can rebuild an extensible index in any of three ways:

- Rebuild all indexes by specifying the `--rebuildAll` option.
- Rebuild the attribute index on which the extensible index is based, by specifying the `-i` option. For example, `-i cn`.

  All indexes based on this attribute are rebuilt, including any extensible indexes that are associated with the attribute.
- Rebuild a specific extensible index by specifying it with the `-i` option. For example, `-i cn.es.lte` or `-i sn.en.sub`.

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/rebuild-index`
- Windows: *INSTANCE_DIR*`\OUD\bat\rebuild-index.bat`

Related Commands

- verify-index
- dsconfig

A.1.3.14 restore

The `restore` command restores a backup of a directory server back end.

Synopsis

`restore` *options*

Description

The `restore` command restores a backup of a directory server back end. Only one back end can be restored at a time. You can use this command to perform a restore operation immediately, or to schedule a restore to run at a later time. For more information, see Configuring Commands As Tasks.

You can restore a back end when the server is offline or schedule a task when the server is online to restore a back end at a later stage. If the server is online, the restore command connects to the server over SSL through the administration connector. For more information about the administration connector, see Managing Administration Traffic to the Server.

Options

The `restore` command accepts an option in either its short form (for example, `-I` *backupID*) or its long form equivalent (for example, `--backupID` *backupID*).

`-d, --backupDirectory` ***path***
Restore using the directory that contains the backup archive. This directory must exist and must contain a backup descriptor file and one or more backups for a given back end. The backup descriptor file is read to obtain information about the available backups and the options used to create them. This is a required option.
You must ensure that the specified path is absolute.

`-I, --backupID` ***backupID***
Specify the backup ID of the backup to be restored. If this option is not provided, the latest backup contained in the backup directory is restored.

`-l, --listBackups`
Display information about the available backups contained in the backup directory. This option causes the command to exit without performing any restore.

`-n, --dry-run`
Verify that the specified backup is valid (that is, ensure that it appears to be a valid archive, and that any hash, signature matches its contents, or both). This option does not actually attempt to restore the backup.

Task Back End Connection Options

Running an online restore requires access to the tasks back end. Access to the tasks back end is provided over SSL through the administration connector. These connection options are used when the restore runs online.

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***hostname***
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

`-j, --bindPasswordFile` ***filename***
Use the bind password in the specified file when authenticating to the directory server.

`-K, --keyStorePath` ***path***
Use the client keystore certificate in the specified path.

`-N, --certNickname` ***nickname***
Use the specified certificate for client authentication.

`-o, --saslOption` ***name=value***
Use the specified options for SASL Authentication.

`-p, --port` ***port***
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

`-P, --trustStorePath` ***path***
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` ***filename***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

`--completionNotify` *emailAddress*
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

`--dependency` *taskId*
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

`--errorNotify` *emailAddress*
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

`--failedDependencyAction` *action*
Specify the action this task will take should one if its dependent tasks fail. The value must be one of `PROCESS`,`CANCEL`,`DISABLE`. If not specified, the backup defaults to `CANCEL`.

`--recurringTask` *schedulePattern*
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

`-t, --start` *startTime*
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 causes the task to be scheduled for immediate execution. When this option is specified, the operation is scheduled to start at the specified time after which this command exits immediately.

Command Input/Output Options

`--noPropertiesFile`
Indicate that a properties file will not be used to get the default command-line options.

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

`-V, --version`
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `restore` command.

Displaying the Backup Information

The following command lists (`-l`) the backup information in the backup descriptor file (`backup.info`) for the directory server. You can use this option to display backup information whether the server is running or stopped.

```
$ restore -l -d /tmp/backup/userRoot
Backup ID:          20081016050258Z
```

```
Backup Date:         16/Oct/2008:09:30:00 +0200
Is Incremental:      false
Is Compressed:       true
Is Encrypted:        true
Has Unsigned Hash:   false
Has Signed Hash:     true
Dependent Upon:      none
```

Restoring a Backup

The following command restores a back end from the backup directory. You can only restore one back end at a time. The server must be stopped before you run this command.

```
$ stop-ds
$ restore -d /tmp/backup/userRoot
[16/Oct/2008:10:32:52 +0200] category=JEB severity=NOTICE msgID=8847445
msg=Restored: 00000000.jdb (size 321954)
```

Restoring an Encrypted Backup

Restoring a hashed or encrypted backup requires a connection to an online server instance, over SSL through the administration connector. When you restore an encrypted backup, you must therefore specify the connection details, including the host, administration port, bind DN and bind password. You must also specify the certificate details for the SSL connection.

The following command restores an encrypted, hashed backup. The self signed certificate is trusted using the `-X` (`--trustAll`) option.

```
$ restore -h localhost -p 4444 -D "cn=directory manager" -j /path/pwd-file -X \
  -d /tmp/backup/userRoot/
Restore task 2008101610403710 scheduled to start immediately
[16/Oct/2008:10:40:38 +0200] severity="NOTICE" msgCount=0 msgID=9896306
 message="The backend userRoot is now taken offline"
[16/Oct/2008:10:40:39 +0200] severity="NOTICE" msgCount=1 msgID=8847445
 message="Restored: 00000000.jdb (size 331434)"
[16/Oct/2008:10:40:40 +0200] severity="NOTICE" msgCount=2 msgID=8847402
 message="The database backend userRoot containing 102 entries has started"
Restore task 2008101610403710 has been successfully completed
```

Scheduling a Restore

Scheduling a restore requires online access to the tasks back end. Access to this back end is provided over SSL through the administration connector. When you schedule a restore, you must therefore specify the connection details, including the host, administration port, bind DN and bind password. You must also specify the certificate details for the SSL connection.

The following command schedules a task to restore the `userRoot` back end at a specific start time by using the `--start` option. The command sends a completion and error notification to `admin@example.com`. The self signed certificate is trusted using the `-X` (`--trustAll`) option.

You can view this scheduled task by using the `manage-tasks` command. For more information, see Configuring Commands As Tasks. You must ensure that the server is running prior to the scheduled restore date and time.

```
$ restore -h localhost -p 4444 -D "cn=directory manager" -j /path/pwd-file -X \
  -d /backup/userRoot --start 20081025121500 --completionNotify admin@example.com \
  --errorNotify admin@example.com
Restore task 2008101610442610 scheduled to start Oct 25, 2008 12:15:00 PM SAST
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `restore` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/restore`
- Windows: *INSTANCE_DIR*`\OUD\bat\restore.bat`

Related Commands

- dbtest
- manage-tasks

A.1.3.15 split-ldif

The `split-ldif` command splits an LDIF file into multiple LDIF files according to a given distribution workflow element. The generated LDIF files are used to populate the partitions of a distribution deployment.

Synopsis

`split-ldif` *options*

Description

The `split-ldif` command splits an LDIF file into multiple LDIF files according to a given distribution workflow element. The data in the LDIF file is split based on the attributes indicated and based on the distribution type defined. The generated LDIF files are then used to populate the partitions. For each partition the split-ldif command creates a partition file as follows:

outputDirectory/*outputFilenamePrefix*-*partitionID*.ldif

Sometimes, the distribution algorithm is not able to determine the partition to which an entry should be sent, either because the entry does not contain all the parameters required by the algorithm, or the required parameters are present but they match no partition. In such a scenario, the output is written to an error file.

All the entries that do not have all the required parameters are written to the following error file:

outputDirectory/*outputFilenamePrefix*-`missingrequired-param`.ldif

All the entries that have the required parameters but whose parameters do not match any configured partition are written to the following error file:

outputDirectory/*outputFilenamePrefix*-`partition-not-found`.ldif

However, for the global index initialization you use the directory containing the files compatible with the global index format. The `split-ldif` command creates one directory per attribute to be indexed, and each directory contains files for initializing the global index.

The global index catalog is populated using the files in the directory created, which do not have a LDIF format. For more information, see gicadm

Options

The `split-ldif` command accepts an option in either its short form (for example, `-i` *ldifFile*) or its long form equivalent (for example, `--ldifFile` *ldifFile*).

`-i, --ldifFile` ***ldifFile***
The name of the LDIF file to split. Global Index Options and Split Options can be used to customize the behavior.

`-l, --listDistributionNames`
Lists the enabled distribution workflow elements from the directory server's configuration.

> **Note:**
>
> The `-l, --listDistributionNames` option lists only the enabled distributions, because you cannot use a disabled distribution to split an ldif file.

Global Index Options

`-x, --index` ***attributeTypeName***
Generates an index file to be used for the global index catalog, for the listed attribute type.

`-c, --onlyCatalog`
Generates only the index file.

Split Options

`-d, --distributionName` ***distributionName***
The name of the distribution workflow element to split the data.

`-p, --forcePartitionId` ***partitionId***
Generates an index file where all the entries are distributed to the same single partition having the listed partitionId.

`-o, --outputDirectory` ***outputDirectory***
The directory where output LDIF files will be generated.

`-O, --outputFilenamePrefix` ***outputFilePrefix***
The prefix of the filename to generate (will contain the partition ID and the.ldif extension).

`-f, --force`
Overwrites generated files that may already exist from previous use.

General Options

`-V, --version`
Display the version information for the directory server.

`-e, --help-examples`
Display examples of the usage.

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to stop or restart the directory server.

Examples

Using split-ldif to Populate a Global Index with One Indexed Attribute

The following command uses an existing database file (`-i`) which it splits into several files, based on the distribution information already defined in the proxy deployment. The command defines the distribution workflow element name (`-d`), the database file (`-i`) to be split, and the attribute to be indexed in the global index files (`-x`). Indicating `-f` will overwrite any existing LDIF files.

You must have deployed a proxy instance with distribution before running this command.

```
$ split-ldif -d "distrib-we" -i database.ldif -x employeenumber -f
```

Assuming, for this example, that your distribution algorithm was numeric, and that you set two partitions with boundaries 1-1000 and 1000-2000. When you run the command above, the following directory and LDIF files are created:

database-1.ldif
This file contains all the entries from database with employee numbers from 1-999, which will be used to populate partition 1.

database-2.ldif
This file contains all the entries from database with employee numbers from 1000-1999, which will be used to populate partition 2.

catalog\employeenumber
This directory contains the global index files for the employee number attribute.

Using split-ldif to Populate a Global Index with Several Indexed Attributes

The following command uses an existing database file (`-i`) which it splits into several files, based on the distribution information already defined in the proxy deployment. The command defines the distribution workflow element name (`-d`), the database file (`-i`) to be split, and the attributes to be indexed in the global index files (`-x`). Indicating `-f` will overwrite any existing LDIF files.

You must have deployed a proxy instance with distribution before running this command.

```
$ split-ldif -d "distrib-we" -i database.ldif \
-x employeenumber -x uid -f
```

Assuming, for this example, that your distribution algorithm was numeric, and that you set two partitions with boundaries 1-50000 and 50000-100001. When you run the command above, the following LDIF files and directories are created:

- `database-1.ldif` - This file contains all the entries from database with employee numbers from 1-49999, which will be used to populate partition 1.
- `database-2.ldif` - This file contains all the entries from database with employee numbers from 50000-100000, which will be used to populate partition 2.
- `catalog\employeenumber` - This directory contains the global index files for the employee number attribute.
- `catalog\uid` - This directory contains the global index files for the uid attribute.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/split-ldif`
- Windows: *INSTANCE_DIR*`\OUD\bat\split-ldif.bat`

Related Commands

- gicadm

A.1.3.16 verify-index

The `verify-index` command validates directory index data.

Synopsis

`verify-index` *options*

Description

The `verify-index` command is used to check the consistency between the index and entry data within the directory server database. This command also provides information about the number of index keys that have reached the index entry limit.

The command checks the following information:

- All entries are properly indexed
- All index data reference entries exist
- Data matches the corresponding index data

Currently, this command is only available for a directory server back end that uses Oracle Berkeley DB Java Edition to store its information. None of the other back end types currently available maintain on-disk indexes. Therefore, there is no need to have any command that can verify index consistency.

Directory administrators can use this command when the directory server is running or stopped. Note, however, that using `verify-index` when the server is running impacts the overall performance of the directory server as well as the command. For example, on a very busy online server, the `verify-index` command could take significantly longer to process compared to running the command on an offline, or stopped, directory server.

To use this command, the `--baseDN` option must be used to specify the base DN of the back end below which to perform the validation.

Options

The `verify-index` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--baseDN` *baseDN*).

Command Options

`-b, --baseDN` ***baseDN***
Specify the base DN for which to perform the verification. The provided value must be a base DN for a back end based on the Berkeley DB Java Edition. This is a required option, and only one base DN may be provided.

-c, --clean
Verify that an index is "clean", which means that all of the entry IDs in all of the index keys refer to entries that actually exist and match the criteria for that index key. If this option is provided, then exactly one index should be specified using the `--index` option. If this option is not given, then the verification process will clean the `id2entry` database (which is a mapping of each entry ID to the actual data for that entry) and ensure that all of the entry contents are properly indexed.

--countErrors
Count the number of errors found during the verification and return that value as the exit code. Values greater than 255 will be returned as `255` due to exit code restrictions.

-i, --index *index*
Specify the name of an index for which to perform the verification. If the `--clean` option is provided, then this argument must be provided exactly once. Otherwise, it may be specified zero or more times. If the option is not provided, then all indexes will be checked. For an attribute index, the index name should be the name of the attribute, and an index must be configured for that attribute in the associated back end. You can also specify the following internal indexes, which are used internally on the server:
`dn2id` - A mapping of entry DNs to their corresponding entry IDs.
`id2children` - A mapping of the entry ID for an entry to the entry IDs of its immediate children.
`id2subtree` - A mapping of the entry ID for an entry to the entry IDs of all of its subordinates.

-v, --verbose
Use verbose mode.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to stop or restart the server.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the `verify-index` command.

Verifying an Index

The following command verifies that the uid index (`-i uid`) under `dc=example,dc=com` (`-b dc=example,dc=com`) is "clean" (`-c`). This "clean" option checks that each entry in the `uid` index maps to an actual database entry with the `uid` attribute.

```
$ verify-index -b dc=example,dc=com -c -i uid

[26/Jul/2007:16:42:31 -0500] category=BACKEND severity=NOTICE msgID=8388709
msg=Checked 150 records and found 0 error(s) in 0 seconds (average rate 331.1/sec)
```

Verifying an Index and Counting Errors

The following command counts the number of discrepancies (`--countErrors`) in the `sn` (surname) index (`-i sn`) under the `dc=example,dc=com` base DN (`-b dc=example,dc=com`):

```
$ verify-index -b dc=example,dc=com -c -i sn --countErrors

[31/Jul/2007:02:23:52 -0500] category=BACKEND severity=NOTICE msgID=8388709 msg=
```

```
Checked 466 records and found 0 error(s) in 0 seconds (average rate 1298.1/sec)
[31/Jul/2007:02:23:52 -0500] category=BACKEND severity=NOTICE msgID=8388710 msg=
Number of records referencing more than one entry: 225
[31/Jul/2007:02:23:52 -0500] category=BACKEND severity=NOTICE msgID=8388711 msg=
Number of records that exceed the entry limit: 0
[31/Jul/2007:02:23:52 -0500] category=BACKEND severity=NOTICE msgID=8388712 msg=
Average number of entries referenced is 2.59/record
[31/Jul/2007:02:23:52 -0500] category=BACKEND severity=NOTICE msgID=8388713 msg=
Maximum number of entries referenced by any record is 150
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 or greater indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/verify-index`
- Windows: *INSTANCE_DIR*`\OUD\bat\verify-index.bat`

Related Commands

- rebuild-index

A.1.3.17 purge-backup

The `purge-backup` command is used to purge backup data from one or more Directory Server back ends that are older than the specified number of days.

Synopsis

`purge-backup` [*options*]

Description

The `purge-backup` command purges backup contents of one or more directory server back ends. The command can perform this operation immediately or at a scheduled time. For more information, see Configuring Commands As Tasks.

Options

The `purge-backup` command accepts an option in either its short form (for example, `-d` *backupDir*) or its long form equivalent (for example, `--backupDirectory` *backupDir*).

`-n, --backendID` ***backendID***
Specify the ID of the backend to be used for purge. This option can be used multiple times in a single command to indicate that the backup data for multiple back ends should be purged. The available backends in the server can be determined by using the `dsconfig list-backends` command.

`-A, --purgeAll`
Purge the back up data for all configured backends. Do not use this option with `--backendID`.

`-k, --purgeDelay` ***purgeDelay***
Purge interval (in days) to consider for purge. Backup sets older than this will be considered for purge. The default value is `120`.

-d, --backupDirectory *backupDir*
Path to the backup directory to purge the backup file(s). If multiple back ends are archived during `purge-backup`, subdirectories should exist for each backend within this directory. If `--purgeAll` option is provided, then this should be the path to the directory which contains all the sub directories for various backends. If a specific backend is provided for purge, then this should be the path to the directory which contains the backup files for the particular backend.

-F, --force
Force purge all backups. By default, the latest backup info set (which includes the latest incremental backup until its parent full backup) will not be deleted. With this option, the latest backup info sets may also be considered for purge based on the other criteria defined above.

> **Note:**
>
> You must bear in mind that this option should be used with caution as it deletes all the backup info sets if they are qualified for purge based on `purgeDelay`.

Task Back End Connection Options

Running an online `purge-backup` requires access to the tasks backend. Access to the tasks backend is provided over SSL through the administration connector. These connection options are used when the `purge-backup` runs online.

-D, --bindDN *bindDN*
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

-h, --hostname *hostname*
Contact the directory server on the specified hostname or IP address. If this option is not provided, a default of `localhost` is used.

-j, --bindPasswordFile *filename*
Use the bind password in the specified file when authenticating to the directory server.

-K, --keyStorePath *path*
Use the client keystore certificate in the specified path.

-N, --certNickname *nickname*
Use the specified certificate for client authentication.

-o, --saslOption *name=value*
Use the specified options for SASL authentication.

-p, --port *port*
Contact the directory server at the specified administration port. If this option is not provided, a default administration port of `4444` is used.

-P, --trustStorePath *path*
Use the client trust store certificate in the specified path. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-u, --keyStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used.

`-U, --trustStorePasswordFile` *filename*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this).

`-X, --trustAll`
Trust all server SSL certificates that the directory server presents. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

Task Scheduling Options

These options are used when you specify that the `purge-backup` should run as a scheduled task.

`--completionNotify` *emailAddress*
Specify the email address of a recipient to be notified when the task completes. This option can be specified more than once in a single command.

`--dependency` *taskId*
Specify the ID of a task upon which this task depends. A task does not start executing until all of its dependencies have completed execution.

`--errorNotify` *emailAddress*
Specify the email address of a recipient to be notified if an error occurs when this task executes. This option can be specified more than once in a single command.

`--failedDependencyAction` *action*
Specify the action that this task will take if one of its dependent tasks fails. The value must be one of `PROCESS`, `CANCEL`, or `DISABLE`. If no value is specified, the default action is `CANCEL`.

`--recurringTask` *schedulePattern*
Indicates that the task is recurring and will be scheduled according to the `schedulePattern`, expressed as a crontab(5) compatible time and date pattern.

`-t, --start` *startTime*
Indicates the date and time at which the operation starts when scheduled as a directory server task expressed in the format `YYYYMMDDhhmmss`. A value of 0 schedules the task for immediate execution. When this option is specified, the operation is scheduled to start at the specified time after which the command exits immediately.

Command Input/Output Options

`--noPropertiesFile`
Indicates that a properties file is not used to obtain the default command-line options.

`--propertiesFilePath` *path*
Specify the path to the properties file that contains the default command-line options.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to purge the back up data.

-V, --version
Display the version information for the directory server and exit rather than attempting to run this command.

Examples

The following examples show how to use the directory server commands.

Purging the Backup Data for All Configured Back Ends

Perform the following steps to purge the backup data for all configured back ends.

1. Display the content of the backup directory to see the subdirectories for each back end:

```
$ ls /tmp/backup
schema tasks userRoot
```

2. Run the following command to examine the contents of the subdirectories to ensure that the backup.info and backup data files are present.

```
$ls /tmp/backup/userRoot/
backup.info  backup-userRoot-20230824134113Z  backup-
userRoot-20230924134151Z
```

3. Run the following command to purge backup data for all directory server back ends (`-A`), in the given backup directory (`-d`), that are older than the specified number of days (`-k 10`).

```
./purge-backup -d /tmp/backup -k 10 -A
```

4. Check the contents of the subdirectories to ensure that the backup data files have been purged and that the backup.info file has been trimmed of the purged sets.

```
$ls /tmp/backup/userRoot/
backup.info  backup-userRoot-20230924134151Z

backup.info contents would look like:

backend_id=userRoot

backup_id=20230924134151Z
backup_date=20230924134155Z
incremental=false
compressed=true
encrypted=false
property.archive_file=backup-userRoot-20230924134151Z
property.last_logfile_size=37964
property.last_logfile_name=00000000.jdb
```

Purging Back Up For a Specific Back End

Use the `-n` option to specify a back end to be backed up. The following command archives the `userRoot` back end only.

```
purge-backup -n userRoot -d /tmp/backup/userRoot -k 10
```

Scheduling a Purge of a Backup

Scheduling a `purge-backup` requires online access to the tasks back end. Access to this back end is provided over SSL through the administration connector. When you schedule a `purge-backup`, you must therefore specify the connection details, including the host, administration port, bind DN and bind password file. You must also specify the certificate details for the SSL connection.

The following command schedules a `purge-backup` of all components (`--purgeAll`) and writes it to the `/tmp/backups` directory (`--backupDirectory`). The start time is specified with the `--start` option. The self signed certificate is trusted using the `-X` (`--trustAll`) option.

```
purge-backup --port 4444 --bindDN "cn=Directory Manager" \
--bindPasswordFile pwd-file -X \
--purgeAll \
--purgeDelay 30 \
--backupDirectory /tmp/backups
--start 20230924131502417
```

You can view this scheduled task by using the `manage-tasks` command. For more information, see Configuring Commands As Tasks.

Exit Codes

An exit code of 0 indicates that the operation completed successfully. An exit code of 1 indicates that an error occurred during processing.

How to Use a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `purge-backup` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

Location

The `purge-backup` command is located at these paths:

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/purge-backup`
- Windows: *INSTANCE_DIR*`\OUD\bat\purge-backup.bat`

Related Commands

- restore
- manage-tasks

A.1.4 LDAP Client Commands

You can review the different options and examples provided by each LDAP client utility.

See Overriding System Default Protocols and Cipher Suites for TLS Communication for overriding ssl protocol and cipher suites for SSL communication using these LDAP client commands.

- ldapcompare
- ldapdelete
- ldapmodify
- ldappasswordmodify
- ldapsearch

A.1.4.1 ldapcompare

The `ldapcompare` command compares LDAP entries.

Synopsis

`ldapcompare` *options*

Description

The `ldapcompare` command issues LDAP compare requests to the directory server. Compare requests can be used to determine whether a given entry or set of entries have a particular attribute-value combination. The only information returned from a successful compare operation is an indication about whether the comparison evaluated to true or false. No other information about the entry is provided.

The syntax of the `ldapcompare` tool on the command-line can take any of these forms:

```
ldapcompare [ options ] attribute:value [ "targetDN" ... | -f DNfile]
ldapcompare [ options ] attribute::base64value [ "targetDN" ... | -f DNfile ]
ldapcompare [ options ] attribute:fileURL [ "targetDN" ... | -f DNfile ]
```

where

- *options* are the command-line options, described in the following section.
- *attribute* is the name of the attribute type, followed by one of the three ways to specify its comparative value. The attribute type name and value string should be enclosed in single quotes (') for the shell.
- *targetDN* is the distinguished name (DN) or list of DNs in which to search for the given attribute and compare its value.
- *DNfile* is a file with a list of DNs, one per line, to search for the given attribute and compare its value.

Options

The `ldapcompare` command accepts an option in either its short form (for example, `-D` *bindDN*) or its long form equivalent (for example, `--bindDN` *bindDN*).

Command Options

`--assertionFilter` *filter*
Perform a search using the LDAP assertion control (as defined in RFC 4528) to indicate that the operation should only be processed if the assertion contained in the provided filter is true.

`-c, --continueOnError`
Continue processing even if an error occurs. This applies when multiple entry DNs have been given either as trailing options or in a file specified with the `--filename` option. If an error occurs while processing a compare request, then the client will continue with the next entry DN if the `--continueOnError` option has been provided, or it will exit with an error if it was not provided.

`-f, --filename` *filename*
Specify the path to a file that contains one or more filters to use when processing the search operation. If there are to be multiple entry DNs, then the file should be structured with one DN per line. All comparisons will be performed using the same connection to the directory server in the order that they appear in the file. If this option is not provided, at least one entry DN must follow the attribute-value assertion. If this option is used, the only trailing option required is the attribute-value assertion. The `--filename` option takes precedence over any DNs provided as additional command-line options. Additional DNs are simply ignored.

`-J, --control` *controloid*`[`*criticality*`[:`*value*`|::`*b64value*`|:<`*fileurl*`]]`
Perform a search with the specified control in search requests sent to the directory server. This option makes it possible to include arbitrary request controls that the client cannot directly support. The value for this option must be in the form:
oid`[:`*criticality*`[:`*value*`|::`*b64value*`|:<`*fileurl*`]]`
The elements of this value include:

- `oid`. Use the OID for the control. For certain types of controls, a text name may be used instead of the numeric OID (for search operations, this includes `managedsait` for the manage DSA IT control). This element is required. Human-readable names can be used in place of the OID to reference controls that do not require values using the `-J` or `control` option. These OID names are the following:
  - `accountusable` or `accountusability`. Use in place of the Account Usability Request Control OID: 1.3.6.1.4.1.42.2.27.9.5.8 (no value)
  - `authzid` or `authorizationidentity`. Use in place of the Authorization Identity Request Control OID: 2.16.840.1.113730.3.4.16 (no value)
  - `effectiverights`. Use in place of the Get Effective Rights Control OID: 1.3.6.1.4.1.42.2.27.9.5.2 (value = authorization ID)
  - `managedsait`. Use in place of the Manage DSA IT Control OID: 2.16.840.1.113730.3.4.2 (no value)
  - `noop` or `no-op`. Use in place of the LDAP No-op Control OID: 1.3.6.1.4.1.4203.1.10.2 (no value)
  - `pwpolicy` or `password policy`. Use in place of the Password Policy Request OID: 1.3.6.1.4.1.42.2.27.8.5.1 (no value)
  - `subtreedelete` or `treedelete`. Use in place of the Subtree Delete Request Control OID: 1.2.840.113556.1.4.805 (no value)

- `criticality`. If `true`, the control should be marked critical (meaning that the directory server should not process the operation unless it can meet the requirements of this control). If `false`, the control should not be marked critical. If this subcommand is not provided, then the control is not marked critical.
- *value*. Specifies the value for the control. Use this form only if the value can be expressed as a string. Do not use this form with either the `::`*b64value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- *b64value*. Specifies the value for the control in base64-encoded form. Do not use this subcommand with either the `:`*value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- *fileurl*. Specifies a URL that references a file from which the value of the control should be taken. Do not use with either the `:`*value* or `::`*b64value* forms. If none of these subcommands is present, then the control will not have a value.

For example, the value `1.3.6.1.4.1.42.2.27.9.5.2:true:dn:uid=dmiller,ou=people,dc=example,dc=com` will include a critical control with an OID of `1.3.6.1.4.1.42.2.27.9.5.2`, marked as critical (true), and with a string value for the authorization ID `dn:uid=dmiller,ou=people,dc=example,dc=com`. Or, you can use the OID names: `effectiverights:true:dn:uid=dmiller,ou=people,dc=example,dc=com`.

`-n, --dry-run`
Run in `no-op` mode. That is, report what should happen but do not actually perform any searches or communicate with the server in any way.

LDAP Connection Options

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***address***
Contact the directory server on the specified host name or IP address. If it is not provided, then a default address of `localhost` will be used.

`-j, --bindPasswordFile` ***bindPasswordFile***
Use the bind password in the specified file when authenticating to the directory server. The option is used for simple authentication, as well as for password-based SASL mechanisms such as CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if no authentication is to be performed. Do not use this option with `--bindPassword`.
SASL is not supported for a proxy server instance.

`-K, --keyStorePath` ***keyStorePath***
Use the client keystore certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option should only be necessary if the client needs to present a certificate to the directory server, for example, when using SASL EXTERNAL authentication.
SASL is not supported for a proxy server instance.

`-N, --certNickName` ***certNickName***
Use the specified certificate for certificate-based client authentication.

`-o, --saslOption` *name=value*
Use the specified option when performing SASL authentication. Multiple SASL options can be provided by using this option multiple times, once for each option.
SASL is not supported for a proxy server instance.

`-p, --port` *port*
Contact the directory server at the specified port. If this option is not provided, then a default port of `389` will be used.

`-P, --trustStorePath` *trustStorePath*
Use the client trust store certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-q, --useStartTLS`
Use the StartTLS Extended Operation when communicating with the directory server. Do not use this option with `--useSSL`.

`-r, --useSASLExternal`
Use the SASL EXTERNAL mechanism for authentication, which attempts to identify the client by using an SSL certificate that it presents to the directory server. If this option is used, then the `--keyStorePath` option must also be provided to specify the path to the client keystore and either the `--useSSL` or the `--useStartTLS` option must be used to establish a secure communication channel with the server.
SASL is not supported for a proxy server instance.

`--trustStorePassword` *trustStorePassword*
Use the password needed to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (which most trust stores do not require). Do not use this option with `--trustStorePasswordFile`.

`-u, --keyStorePasswordFile` *keyStorePasswordFile*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePassword`.

`-U, --trustStorePasswordFile` *trustStorePasswordFile*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this). Do not use this option with `--trustStorePassword`.

`-V, --ldapVersion` *version*
Set the LDAP protocol version that the client should use when communicating with the directory server. The value must be either `2` (for LDAPv2 communication) or `3` (for LDAPv3). If this option is not provided, then the client will use LDAPv3.

`-w, --bindPassword` *bindPassword*
Use the bind password when authenticating to the directory server. This option can be used for simple authentication as well as password-based SASL mechanisms. Do not use this option with `--bindPasswordFile`. To prompt for the password, type `-w -`.
SASL is not supported for a proxy server instance.

`-W, --keyStorePassword` *keyStorePassword*
Use the password needed to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePasswordFile`.

`-X, --trustAll`
Trust any certificate that the directory server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

`-Z, --useSSL`
Use Secure Sockets Layer when communicating with the directory server. If SSL is to be used, then the `--port` option should be used to specify the server's secure port.

Command Input/Output Options

`--noPropertiesFile`
Indicate that a properties file will not be used to get the default command-line options.

`--propertiesFilePath` *propertiesFilePath*
Specify the path to the properties file that contains the default command-line options.

`-v, --verbose`
Run in verbose mode, displaying process and diagnostic information on standard output.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to run the command.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the `ldapcompare` command.

Comparing an Entity for Group Membership

The following command specifies the host name (`-h`) that is connected to port `1389` (`-p`) and verifies if an employee (`uid=scarter`) is a member of a group (`cn=Accounting Managers`).

```
$ ldapcompare -h hostname -p 1389 \
"uniquemember:uid=scarter,ou=People,dc=example,dc=com" \
"cn=Accounting Managers,ou=groups,dc=example,dc=com"

Comparing type uniquemember with value uid=scarter,ou=People,dc=example,dc=com
in entry cn=Accounting Managers,ou=groups,dc=example,dc=com
Compare operation returned true for entry
cn=Accounting Managers,ou=groups,dc=example,dc=com
```

Comparing an Attribute Value to an Entry

The following command specifies the hostname (`-h`) that is connected to port `1389` (`-p`) and verifies if an attribute (`ou=Accounting`) is present in an entity's (`cn=Sam Carter`) record.

```
$ ldapcompare -h hostname -p 1389 "ou:Accounting" \
"uid=scarter,ou=People,dc=example,dc=com"
```

```
Comparing type ou with value Accounting in entry uid=scarter,ou=People,dc=example,dc=com
Compare operation returned true for entry uid=scarter,ou=People,dc=example,dc=com
```

Using ldapcompare with Server Authentication

The following command uses server authentication, specifies the host name (`-h`), SSL port (`-p`), base DN (`-b`), the bind DN (`-D`), the bind password (`-w`), trust store file path (`-P`), and checks if the attribute is present in the entry. For Windows platforms, use the path where your trust store file resides (for example, `-P \temp\certs\cert.db`).

```
$ ldapcompare -h hostname -p 1636 -D "cn=Directory Manager" \
-j pwd-file -P /home/kwinters/certs/cert.db \
'givenname:Sam' "uid=scarter,ou=People,dc=example,dc=com"

Comparing type givenname with value Sam in entry uid=scarter,ou=People,dc=example,dc=com
Compare operation returned true for entry uid=scarter,ou=People,dc=example,dc=com
```

Using ldapcompare with Client Authentication

The following command uses client authentication with the compare. The command uses SSL (`-Z`) with the SSL port (`-p`), specifies the trust store file path (`-P`), the certificate nickname (`-N`), the keystore file path (`-K`), the keystore password (`-W`) and checks if the entity's given name `givenname=Sam` is present in the entry. For Windows platforms, use the path where your trust store file resides (for example, `-P \temp\certs\cert.db`) and where the path where your keystore file resides (`-K \temp\security\key.db`).

```
$ ldapcompare -h hostname -p 1636 -Z \
-P /home/kwinters/security/cert.db -N "kwcert" \
-K /home/kwinters/security/key.db -W KeyPassword \
'givenname:Sam' "uid=scarter,ou=People,dc=example,dc=com"

Comparing type givenname with value Sam in entry uid=scarter,ou=People,dc=example,dc=com
Compare operation returned true for entry uid=scarter,ou=People,dc=example,dc=com
```

Exit Codes

An exit code of 6 indicates that the comparison is successful. An exit code of 5 indicates that the comparison is unsuccessful. Any other exit code indicates that an error occurred during processing.

How to Use a CLI Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `ldapcompare` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see Using a Properties File With Server Commands.

The following options can be stored in a properties file:

- `assertionFilter`
- `bindDN`
- `bindPassword`
- `bindPasswordFile`
- `certNickname`
- `continueOnError`
- `control`

- dry-run
- filename
- hostname
- keyStorePassword
- keyStorePasswordFile
- keyStorePath
- ldapVersion
- port
- saslOption
- trustAll
- trustStorePassword
- trustStorePasswordFile
- trustStorePath
- useSASLExternal
- useSSL
- useStartTLS
- verbose

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
ldapcompare.ldapport=12345
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldapcompare`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldapcompare.bat`

Related Commands

- ldapdelete
- ldapmodify
- ldappasswordmodify
- ldapsearch

A.1.4.2 ldapdelete

The `ldapdelete` command issues LDAP delete requests to the directory server to remove entries.

Synopsis

`ldapdelete` [*option*] [*DN*]

Description

The `ldapdelete` command issues LDAP delete requests to the directory server to remove entries. Unless the `--filename` option is given, an entry DN must be given as the only trailing option to specify which entry should be removed.

Before You Begin

Many UNIX or Linux operating systems provide an installed version of common LDAP client commands, such as `ldapsearch`, `ldapmodify`, and `ldapdelete` in the `/usr/bin` directory. You can check if a version is on your system by entering the command: `which ldapdelete`. If the command returns a value (seen below), you must update your `$PATH` to the *INSTANCE_DIR*`/OUD/bin` directory or create an alias to the directory server instance.

```
$ which ldapdelete (UNIX/Linux)
/usr/bin/ldapdelete
```

Options

The `ldapdelete` command accepts an option in either its short form (for example, `-D` *bindDN*) or its long form equivalent (for example, `--bindDN` *bindDN*).

Command Options

`-c, --continueOnError`
Continue processing even if an error occurs. This operation applies when multiple entry DNs have been given either as trailing options or in a file specified with the `--filename` option. If an error occurs while processing a compare request, then the client will continue with the next entry DN if the `--continueOnError` option has been provided, or it will exit with an error if that option was not provided.

`-f, --filename` ***filename***
Specify the path to a file that contains one or more filters to use when processing the search operation. If there are multiple entry DNs, then the file should be structured with one DN per line. If this option is used, then do not add any trailing options. The DN of the entry to remove should be the only trailing option.

`-J, --control` ***controloid*[:*criticality*[:*value*|::*b64value*|:<*fileurl*]]**
Perform a search with the specified control in search requests sent to the directory server. This option makes it possible to include arbitrary request controls that the client cannot directly support. The value for this option must be in the form:
oid[:*criticality*[:*value*|::*b64value*|:<*fileurl*]]
The elements of this value include:

- **oid.** Use the OID for the control. For certain types of controls, a text name may be used instead of the numeric OID (for search operations, this includes `managedsait` for the manage DSA IT control). This element is required. Human-readable names can be used in place of the OID to reference controls that do not require values using the -J or `control` option. These OID names are the following:

  `accountusable` or `accountusability` — Use in place of the Account Usability Request Control OID:1.3.6.1.4.1.42.2.27.9.5.8 (no value).

  `authzid` or `authorizationidentity` — Use in place of the Authorization Identity Request Control OID: 2.16.840.1.113730.3.4.16 (no value).

`effectiverights` — Use in place of the Get Effective Rights Control OID: 1.3.6.1.4.1.42.2.27.9.5.2 (value = authorization ID).

`managedsait` — Use in place of the Manage DSA IT Control OID: 2.16.840.1.113730.3.4.2 (no value).

`noop` or `no-op` — Use in place of the LDAP No-op Control OID: 1.3.6.1.4.1.4203.1.10.2 (no value).

`pwpolicy` or `password policy` — Use in place of the Password Policy Request Control OID: 1.3.6.1.4.1.42.2.27.8.5.1 (no value).

`subtreedelete` or `treedelete` — Use in place of the Subtree Delete Request Control OID: 1.2.840.113556.1.4.805 (no value).

- **criticality.** If `true`, the control should be marked critical (meaning that the directory server should not process the operation unless it can meet the requirements of this control). If `false`, the control should not be marked critical. If this subcommand is not provided, then the control is not marked critical.
- **value.** Specifies the value for the control. This form should only be used if the value can be expressed as a string. Do not use this form with either the `::`*b64value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **b64value.** Specifies the value for the control in base64-encoded form. Do not use this subcommand with either the `:`*value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **fileurl.** Specifies a URL that references a file from which the value of the control should be taken. Do not use with either the `:`*value* or `::`*b64value* forms. If none of these subcommands is present, then the control will not have a value.

  For example, the value `1.3.6.1.4.1.42.2.27.9.5.2:true:dn:uid=dmiller,ou=people,dc=example,dc=com` will include a critical control with an OID of `1.3.6.1.4.1.42.2.27.9.5.2`, marked as critical (true), and with a string value for the authorization ID `dn:uid=dmiller,ou=people,dc=example,dc=com`. Or, you can use the OID names: `effectiverights:true:dn:uid=dmiller,ou=people,dc=example,dc=com`.

`-n, --dry-run`
Run in `no-op` mode. That is, report what should happen but do not actually perform any searches or communicate with the server in any way.

`-x, --deleteSubtree`
Delete the specified entry and all entries below it.

LDAP Connection Options

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication and is not required if SASL authentication is to be used. The default value for this option is `cn=Directory Manager`.

`-h, --hostname` ***address***
Contact the directory server on the specified host name or IP address. If it is not provided, then a default address of `localhost` will be used.

`-j, --bindPasswordFile` ***bindPasswordFile***
Use the bind password in the specified file when authenticating to the directory server. The option is used for simple authentication, as well as for password-based SASL mechanisms

such as CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if no authentication is to be performed. Do not use this option with `--bindPassword`.
SASL is not supported for a proxy server instance.

`-K, --keyStorePath` ***keyStorePath***
Use the client keystore certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option should only be necessary if the client needs to present a certificate to the directory server, for example, when using SASL EXTERNAL authentication.
SASL is not supported for a proxy server instance.

`-N, --certNickName` ***certNickName***
Use the specified certificate for certificate-based client authentication.

`-o, --saslOption` ***name*** **=** ***value***
Use the specified option when performing SASL authentication. Multiple SASL options can be provided by using this option multiple times, once for each option. See Using SASL Authentication for more information.
SASL is not supported for a proxy server instance.

`-p, --port` ***port***
Contact the directory server at the specified port. If this option is not provided, then a default port of `389` will be used.

`-P, --trustStorePath` ***trustStorePath***
Use the client trust store certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-q, --useStartTLS`
Use the StartTLS Extended Operation when communicating with the directory server. Do not use this option with `--useSSL`.

`-r, --useSASLExternal`
Use the SASL EXTERNAL mechanism for authentication, which attempts to identify the client by using an SSL certificate that it presents to the directory server. If this option is used, then the `--keyStorePath` option must also be provided to specify the path to the client keystore and either the `--useSSL` or the `--useStartTLS` option must be used to establish a secure communication channel with the server.
SASL is not supported for a proxy server instance.

`--trustStorePassword` ***trustStorePassword***
Use the password needed to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (which most trust stores do not require). Do not use this option with `--trustStorePasswordFile`.

`-u, --keyStorePasswordFile` ***keyStorePasswordFile***
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePassword`.

`-U, --trustStorePasswordFile` ***trustStorePasswordFile***
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a

password to access its contents (most trust stores do not require this). Do not use this option with `--trustStorePassword`.

`-V, --ldapVersion` *version*
Set the LDAP protocol version that the client should use when communicating with the directory server. The value must be either `2` (for LDAPv2 communication) or `3` (for LDAPv3). If this option is not provided, then the client will use LDAPv3.

`-w, --bindPassword` *bindPassword*
Use the bind password when authenticating to the directory server. This option can be used for simple authentication as well as password-based SASL mechanisms. Do not use this option with `--bindPasswordFile`. To prompt for the password, type `-w -`.
SASL is not supported for a proxy server instance.

`-W, --keyStorePassword` *keyStorePassword*
Use the password needed to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePasswordFile`.

`-X, --trustAll`
Trust any certificate that the directory server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

`-Z, --useSSL`
Use Secure Sockets Layer when communicating with the directory server. If SSL is to be used, then the `--port` option should be used to specify the server's secure port.

Command Input/Output Options

`--noPropertiesFile`
Indicate that a properties file will not be used to get the default command-line options.

`--propertiesFilePath` *propertiesFilePath*
Specify the path to the properties file that contains the default command-line options.

`-v, --verbose`
Run in verbose mode, displaying process and diagnostic information on standard output.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to run the command.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the `ldapdelete` command.

Deleting an Entry from the Command Line

The following command specifies the host name (`-h`), the port (`-p`), the bind DN (`-D`), the bind password (`-w`), and deletes a single entry:

```
$ ldapdelete -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
"uid=mgarza,ou=People,dc=example,dc=com"
```

Deleting Multiple Entries by Using a DN File

The following file contains a list of DN's for deletion. The file must list each DN on a separate line.

```
uid=mgarza,ou=People,dc=example,dc=com
uid=wsmith,ou=People,dc=example,dc=com
uid=jarrow,ou=People,dc=example,dc=com
uid=mbean,ou=People,dc=example,dc=com
```

The following command specifies the host name (`-h`), the port (`-p`), the bind DN (`-D`), and the bind password (`-w`), and reads the entries in a file for deletion. If an error occurs, the command continues (`-c`) to the next search item. For Windows platforms, use the path where the deletion file resides (for example, `-f \temp\delete.ldif`):

```
$ ldapdelete -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-c -f /usr/local/delete.ldif
```

Deleting Entries by Using Server Authentication

The following command uses server authentication to delete an entry. The command specifies the host name (`-h`), SSL port (`-p`), bind DN (`-D`), the bind password (`-w`), trust store file path (`-P`), and LDIF file (`-f`) that contains the deletes. If an error occurs, the command continues (`-c`) to the next search item. For Windows platforms, use the path where the deletion file resides (for example, `-f \temp\delete.ldif`) and the file where the trust store password resides (for example, `-P \temp\certs\cert.db`):

```
$ ldapdelete -h hostname -p 1636 -c -f /usr/local/delete.ldif \
-D "cn=Directory Manager" -j pwd-file \
-P /home/kwinters/certs/cert.db
```

Deleting Entries by Using Client Authentication

The following command uses client authentication to perform a delete option. The command uses SSL (`-Z`) with the SSL port (`-p`), specifies the trust store file path (`-P`), the certificate nickname (`-N`), the keystore file path (`-K`), the keystore password (`-W`) and the LDIF file (`-f`) that contains the deletions. If an error occurs, the command continues (`-c`) to the next search item. For Windows platforms, use the path where the deletion file resides (for example, `-f \temp\delete.ldif`), the file where the trust store password resides (for example, `-P \temp\certs\cert.db`), and the file where the keystore password resides (for example, `-K \temp\security\key.db`).

```
$ ldapdelete -h hostname -p 1636 -c -f /usr/local/delete.ldif \
-Z -P /home/kwinters/security/cert.db -N "kwcert" \
-K /home/kwinters/security/key.db -W keypassword
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

How to Use a CLI Properties File

The directory server supports the use of a properties file that passes in any default option values used with the `ldapdelete` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. See Using a Properties File With Server Commands for more information.

The following options can be stored in a properties file:

- `bindDN`
- `bindPassword`
- `bindPasswordFile`
- `certNickname`
- `continueOnError`
- `control`
- `deleteSubtree`
- `dry-run`
- `filename`
- `hostname`
- `keyStorePassword`
- `keyStorePasswordFile`
- `keyStorePath`
- `ldapVersion`
- `port`
- `saslOption`

  SASL is not supported for a proxy server instance
- `trustAll`
- `trustStorePassword`
- `trustStorePasswordFile`
- `trustStorePath`
- `useSASLExternal`

  SASL is not supported for a proxy server instance.
- `useSSL`
- `useStartTLS`
- `verbose`

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
ldapdelete.ldapport=12345
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldapdelete`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldapdelete.bat`

Related Commands

- ldapcompare

- ldapmodify
- ldappasswordmodify
- ldapsearch

A.1.4.3 ldapmodify

The `ldapmodify` command modifies directory entries.

Synopsis

`ldapmodify` [*options*] [*filter*] [*attributes*]

Description

The `ldapmodify` command can be used to perform LDAP modify, add, delete, and modify DN operations in the directory server. The operations to perform in the directory server should be specified in LDIF change format, as described in RFC 2849 (`http://www.ietf.org/rfc/rfc2849.txt`). This change syntax uses the `changetype` keyword to indicate the type of change.

An `add` change record is straightforward, because it is a complete entry in LDIF form with a `changetype` value of `add`. For example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: john.doe
givenName: John
sn: Doe
cn: John Doe
mail: john.doe@example.com
userPassword: password
```

A `delete` change record is even simpler than an `add` change record. The `add` record consists of a line with the entry DN followed by another line with a `changetype` of `delete`. For example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: delete
```

The `modify` change record is the most complex operation, because of the number of variants. The `modify` change records all start with the entry DN followed by a `changetype` of `modify`. The next line consists of either `add`, `delete`, or `replace` followed by an attribute name indicating what modification will be and to which attribute. The change record may optionally be followed by one or more lines containing the attribute name followed by a value to use for the modification (that is, a value to add to that attribute, remove from that attribute, or use to replace the existing set of values). Multiple attribute changes can be made to an entry in the same `modify` operation by separating changes with a line containing only a dash, starting the next line with a new `add`, `delete`, or `replace` tag followed by a colon and the next attribute name, and then setting of values for that attribute. For example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: This is the new description for John Doe
-
```

```
add: mailAlternateAddress
mailAlternateAddress: jdoe@example.com
```

Modify DN change records should always contain the `newRDN` and `deleteOldRDN` elements and can optionally contain the `newSuperior` component to specify a new parent for the target entry. For example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: moddn
newRDN: uid=jdoe
deleteOldRDN: 1
```

If no arguments are provided to the `ldapmodify` command, it attempts to interact with a Directory Server instance using an unauthenticated connection using the loopback address on port 389, and information about the changes to request will be read from standard input. This is unlikely to succeed, as it will almost certainly be necessary to at least provide arguments that will be used to specify how to authenticate to the server.

Before You Begin

Many UNIX and Linux operating systems provide an installed version of common LDAP client commands, such as `ldapsearch`, `ldapmodify`, and `ldapdelete` in the `/usr/bin` directory. You can check if a version is on your system by entering the command: `which ldapmodify`. If the command returns a value (seen below), you must update your `$PATH` to *INSTANCE_DIR*`/OUD/bin` or create an alias to the directory server instance.

```
$ which ldapmodify (Unix/Linux)
/usr/bin/ldapmodify
```

Options

The `ldapmodify` command accepts an option in either its short form (for example, `-D` *bindDN*) or its long form equivalent (for example, `--bindDN` *bindDN*).

Command Options

`-a, --defaultAdd`
Add entries. Treat records with no `changetype` element as an add request. This option can be used to add entries from a standard LDIF file that does not contain information in the LDIF change format.

`--assertionFilter` ***filter***
Perform a search using the LDAP assertion control (as defined in RFC 4528 (`http://www.ietf.org/rfc/rfc4528.txt`)) to indicate that the operation should only be processed if the assertion contained in the provided filter is true.

`-c, --continueOnError`
Continue processing even if an error occurs. Use this option when using multiple search filters in a file `--filename`. If an error occurs during processing, the directory server will continue processing the next search filter. Otherwise the command will exit before all searches have been completed.

`-f, --filename` ***filename***
Read modifications from the specified file containing one or more filters to use during the modify operation. The records in the LDIF file should be in the LDIF change format (that is, including the `changetype` element). If the LDIF file only contains entries that should be added to the directory server, then the file can be used with the `--defaultAdd` option even if the

entries do not have a `changetype` element. The provided file can contain multiple changes if there is at least one blank line between change records.
If this option is not provided, then the `ldapmodify` command will attempt to read change information from standard input. This makes it possible to have the change records either provided interactively by the target user on the command line or piped into the command from some other source.

`-J, --control` *controloid*[:*criticality*[:*value*|::*b64value*|:<*fileurl*]]
Perform a search with the specified control in search requests sent to the directory server. This option makes it possible to include arbitrary request controls that the client cannot directly support. The value for this option must be in the form:
oid[: *criticality*[:*value*|::*b64value*|:<*fileurl*]]
The elements of this value include:

- **oid.** Use the OID for the control. For certain types of controls, a text name may be used instead of the numeric OID (for search operations, this includes `managedsait` for the manage DSA IT control). This element is required. Human-readable names can be used in place of the OID to reference controls that do not require values using the -J or `control` option. These OID names are the following:

  `accountusable` or `accountusability` — Use in place of the Account Usability Request Control OID: 1.3.6.1.4.1.42.2.27.9.5.8 (no value).

  `authzid` or `authorizationidentity` — Use in place of the Authorization Identity Request Control OID: 2.16.840.1.113730.3.4.16 (no value).

  `effectiverights` — Use in place of the Get Effective Rights Control OID: 1.3.6.1.4.1.42.2.27.9.5.2 (value = authorization ID).

  `managedsait` — Use in place of the Manage DSA IT Control OID: 2.16.840.1.113730.3.4.2 (no value).

  `noop` or `no-op` — Use in place of the LDAP No-op Control OID: 1.3.6.1.4.1.4203.1.10.2 (no value).

  `pwpolicy` or `password policy` — Use in place of the Password Policy Request Control OID: 1.3.6.1.4.1.42.2.27.8.5.1 (no value).

  `subtreedelete` or `treedelete` — Use in place of the Subtree Delete Request Control OID: 1.2.840.113556.1.4.805 (no value).

- **criticality.** If `true`, the control should be marked critical (meaning that the directory server should not process the operation unless it can meet the requirements of this control). If `false`, the control should not be marked critical. If this subcommand is not provided, then the control is not marked critical.
- **value.** Specifies the value for the control. Use this form only if the value can be expressed as a string. Do not use this form with either the `::`*b64value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **b64value.** Specifies the value for the control in base64-encoded form. Do not use this subcommand with either the `:`*value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **fileurl.** Specifies a URL that references a file from which the value of the control should be taken. Do not use with either the `:`*value* or `::`*b64value* forms. If none of these subcommands is present, then the control will not have a value.

For example, the value `1.3.6.1.4.1.42.2.27.9.5.2:true:dn:uid=dmiller,ou=people,dc=example,dc=com` will include a critical control with an OID of `1.3.6.1.4.1.42.2.27.9.5.2`, marked as critical (true), and with a string value for the authorization ID `dn:uid=dmiller,ou=people,dc=example,dc=com`. Or, you can use the OID names: `effectiverights:true:dn:uid=dmiller,ou=people,dc=example,dc=com`.

`-n, --dry-run`
Run in `no-op` mode. That is, report what should happen but do not actually perform any searches or communicate with the server in any way.

`--postReadAttributes` ***attrList***
Use the LDAP ReadEntry Post-read Control (as defined in RFC 4527 (`http://www.ietf.org/rfc/rfc4527.txt`)) to indicate that the directory server should return a copy of the target entry as it was immediately after the update. This is only applicable for add, modify, and modify DN operations. The value for this option should be a comma-separated list of the attributes to include in the representation of the pre-read entry. The same conventions apply to this list as for the list of attributes to return in the `ldapsearch` command (that is, it is possible to use `*` for all user attributes, `+` for all operational attributes, `@`*ocname* for all attributes in the specified objectclass, and so on). If no attributes are specified (signified with empty quotes), then all user attributes will be returned.

`--preReadAttributes` ***attrList***
Use the LDAP ReadEntry Pre-read Control (as defined in RFC 4527 (`http://www.ietf.org/rfc/rfc4527.txt`)) to indicate that the directory server should return a copy of the target entry as it was immediately before the update. This is only applicable for delete, modify, and modify DN operations. The value for this option should be a comma-separated list of the attributes to include in the representation of the pre-read entry. The same conventions apply to this list as for the list of attributes to return in the `ldapsearch` command (that is, it is possible to use `*` for all user attributes, `+` for all operational attributes, `@`*ocname* for all attributes in the specified objectclass, and so on). If no attributes are specified (signified with empty quotes), then all user attributes will be returned.

`-Y, --proxyAs` ***authzID***
Use the Proxied Authorization Control to specify the identity of the user for whom the operations should be performed. This will use version 2 of the Proxied Authorization Control as defined in RFC 4370 (`http://www.ietf.org/rfc/rfc4370.txt`). The value of the option should be an authorization ID in the form `dn:` followed by the DN of the target user (for example, `dn:uid=john.doe,ou=People,dc=example,dc=com`), or `u:` followed by the user name (for example, `u:john.doe`). If this option is not provided, then proxied authorization will not be used.

LDAP Connection Options

`-D, --bindDN` ***bindDN***
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication. The default value for this option is `cn=Directory Manager`. It is not required when using SASL authentication or if no authentication is to be performed.

`-E, --reportAuthzID`
Use the authorization identity request control (as defined in RFC 3829 (`http://www.ietf.org/rfc/rfc3829.txt`)) in the bind request so that the directory server returns the corresponding authorization ID to the client when authentication has completed. (The line containing the authorization ID will be prefixed with a `#` character, making it a comment if the output is to be interpreted as an LDIF.)

`-h, --hostname` *address*
Contact the directory server on the specified host name or IP address. If it is not provided, then a default address of `localhost` will be used.

`-j, --bindPasswordFile` *bindPasswordFile*
Use the bind password in the specified file when authenticating to the directory server. The option is used for simple authentication, as well as for password-based SASL mechanisms such as CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if no authentication is to be performed. Do not use this option with `--bindPassword`.
SASL is not supported for a proxy server instance.

`-K, --keyStorePath` *keyStorePath*
Use the client keystore certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option should only be necessary if the client needs to present a certificate to the directory server, for example, when using SASL EXTERNAL authentication.
SASL is not supported for a proxy server instance.

`-N, --certNickName` *certNickName*
Use the specified certificate for certificate-based client authentication.

`-o, --saslOption` *name* = *value*
Use the specified option when performing SASL authentication. Multiple SASL options can be provided by using this option multiple times, once for each option. For information about using SASL authentication in clients, see Configuring SASL Authentication.
SASL is not supported for a proxy server instance.

`-p, --port` *port*
Contact the directory server at the specified port. If this option is not provided, then a default port of `389` will be used.

`-P, --trustStorePath` *trustStorePath*
Use the client trust store certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-q, --useStartTLS`
Use the StartTLS extended operation when communicating with the directory server. Do not use this option with `--useSSL`.

`-r, --useSASLExternal`
Use the SASL EXTERNAL mechanism for authentication, which attempts to identify the client by using an SSL certificate that it presents to the directory server. If this option is used, then the `--keyStorePath` option must also be provided to specify the path to the client keystore and either the `--useSSL` or the `--useStartTLS` option must be used to establish a secure communication channel with the server.
SASL is not supported for a proxy server instance.

`--trustStorePassword` *trustStorePassword*
Use the password needed to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (which most trust stores do not require). Do not use this option with `--trustStorePasswordFile`.

-u, --keyStorePasswordFile *keyStorePasswordFile*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePassword`.

-U, --trustStorePasswordFile *trustStorePasswordFile*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this). Do not use this option with `--trustStorePassword`.

-V, --ldapVersion *version*
Set the LDAP protocol version that the client should use when communicating with the directory server. The value must be either `2` (for LDAPv2 communication) or `3` (for LDAPv3). If this option is not provided, then the client will use LDAPv3.

-w, --bindPassword *bindPassword*
Use the bind password when authenticating to the directory server. This option can be used for simple authentication as well as password-based SASL mechanisms. Do not use this option with `--bindPasswordFile`. To prompt for the password, type `-w -`.
SASL is not supported for a proxy server instance.

-W, --keyStorePassword *keyStorePassword*
Use the password needed to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePasswordFile`.

-X, --trustAll
Trust any certificate that the directory server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

-Z, --useSSL
Use SSL when communicating with the directory server. If SSL is to be used, then the `--port` option should be used to specify the server's secure port.

Command Input/Output Options

--noPropertiesFile
Indicate that a properties file will not be used to get the default command-line options.

--propertiesFilePath *propertiesFilePath*
Specify the path to the properties file that contains the default command-line options.

-v, --verbose
Run in verbose mode, displaying process and diagnostic information on standard output.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to run the command.

-V, --version
Display the version information for the directory server.

Examples

The following examples show how to use the `ldapmodify` command.

Adding an Entry

The following LDIF file contains an entry for an employee:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
cn: Marcia Garza
sn: Garza
givenName: Marcia
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
ou: People
```

The following command specifies the host name (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), reads the modifications from the file (`-f`) and adds the entry (`-a`) to the database. For Windows platforms, specify the path to your LDIF file (for example, `-f \temp\add_entry.ldif`).

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-a -f /usr/local/add_entry.ldif
```

Adding an Attribute to an Entry

The following LDIF file modifies an entry by adding a `telephonenumber` attribute:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: +1 408 555 8283
```

The following command specifies the host name (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), reads the modifications from the file (`-f`) and adds an attribute to the entry. For Windows platforms, specify the path to your LDIF file (for example,

`-f \temp\add_attribute.ldif`).

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-f /usr/local/add_attribute.ldif
```

Modifying the Value of an Attribute

The following LDIF file modifies the value of the `telephonenumber` attribute:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 6456
```

The following command specifies the hostname (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), reads the modifications from the file (`-f`) and modifies the attribute's value. For Windows-platforms, specify the path to your LDIF file (for example, `-f \temp\modify_attribute.ldif`).

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-f /usr/local/modify_attribute.ldif
```

Modifying Multiple Attributes

The following LDIF file contains multiple modifications to an entry:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 6465
-
add: facsimiletelephonenumber
facsimiletelephonenumber: +1 408 222 4444
-
add: l
l: Sunnyvale
```

The following command specifies the host name (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), reads the modifications from the file (`-f`) and processes the changes to the database. For Windows platforms, specify the path to your LDIF file (for example,`-f \temp\mod_attribute.ldif`):

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-f /usr/local/mod_attribute.ldif
```

Deleting an Attribute from the Command Line

The following command specifies the host name (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), and deletes the `facsimiletelephonenumber` attribute for an entry. Because the command is run from the command line, enter the `dn`, `changetype`, modification operation, and then press Control-D (UNIX, Linux) or Control-Z (Windows) to process it:

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
delete: facsimiletelephonenumber
(Press Control-D for Unix, Linux)
(Press Control-Z for Windows)
```

Deleting an Entry from the Command Line

The following command specifies the hostname (`-h`), port (`-p`), bind DN (`-D`), bind password (`-w`), and deletes the entry. Because the command is run from the command line, enter the `dn`, `changetype`, and then press Control-D (UNIX, Linux) or Control-Z (Windows) to process it:

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: delete
(Press Control-D for Unix, Linux)
(Press Control-Z for Windows)
```

Using ldapmodify with Server Authentication

The following command uses the `-P` SSL option to perform a modify with server authentication. The command specifies the host name (`-h`), SSL port (`-p`), base DN (`-b`), the bind DN (`-D`), the bind password (`-w`), trust store file path (`-P`), and LDIF file (`-f`) that contains the changes. For Windows platforms, specify the paths for the modification file (for example, `-f \temp\myldif.ldif`) and trust store file (for example, `-P \temp\certs\cert.db`):

```
$ ldapmodify -h hostname -p 1636 -f /home/local/myldif.ldif \
-D "cn=Directory Manager" -j pwd-file \
-P /home/scarter/certs/cert.db
```

Using ldapmodify with Client Authentication

The following command uses the `-P` SSL option to perform a modify using client authentication. The command uses SSL (`-Z`) with the SSL port (`-p`) and specifies the trust store file path (`-P`), the certificate nickname (`-N`), the keystore file path (`-K`), the keystore password (`-W`) and the LDIF file (`-f`) that contains the changes. For Windows platforms, specify the paths for the modification file (for example, `-f \temp\myldif.ldif`), trust store file (for example, `-P \certs\cert.db`), and the keystore file (for example, `-K \security\key.db`):

```
$ ldapmodify -h hostname -p 1636 -f /home/local/myldif.ldif \
-Z -P /home/scarter/security/cert.db -N "sccert" \
-K /home/scarter/security/key.db -W keypassword
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

How to Use a CLI Properties File

The directory server supports the use of a properties file that passes in any default option values used with the `ldapmodify` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. See Using a Properties File With Server Commands for more information.

- `assertionFilter`
- `bindDN`
- `bindPassword`
- `bindPasswordFile`
- `certNickname`
- `continueOnError`
- `control`
- `dry-run`
- `filename`
- `hostname`
- `keyStorePassword`
- `keyStorePasswordFile`
- `keyStorePath`
- `ldapVersion`
- `port`
- `postReadAttributes`
- `preReadAttributes`
- `proxyAs`
- `reportAuthzID`
- `saslOption`

  SASL is not supported for a proxy server instance.

- `trustAll`
- `trustStorePassword`
- `trustStorePasswordFile`
- `trustStorePath`
- `useSASLExternal`

  SASL is not supported for a proxy server instance.
- `useSSL`
- `useStartTLS`
- `verbose`

The following options can be stored in a properties file:

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
ldapmodify.ldapport=12345
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldapmodify`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldapmodify.bat`

Related Commands

- ldapcompare
- ldapdelete
- ldappasswordmodify
- ldapsearch

A.1.4.4 ldappasswordmodify

The `ldappasswordmodify` command modifies LDAP passwords.

Synopsis

`ldappasswordmodify` *options*

Description

The `ldappasswordmodify` command can be used to change or reset user passwords with the LDAP password modify extended operation as defined in RFC 3062 (`http://www.ietf.org/rfc/rfc3062.txt`).

Using this mechanism for changing user passwords offers several benefits over a simple LDAP modify operation targeted at the password attribute, including the following:

- Changing one's own password. The command allows users to change their own password even after it has expired, if this capability is allowed in their password policy.

- Supplying clear-text password. The command provides a mechanism for supplying the clear-text version of the current password for further validation of the user's identity.
- Using authorization ID. When changing a user's password, the user can be specified by using an authorization ID (prefixed by `dn:` or `u:`) in addition to a full DN.
- Generating passwords. If a new password is not provided, then the server can generate one for the user if this capability is allowed in their password policy.

Options

The `ldappasswordmodify` command accepts an option in either its short form (for example, `-D` *bindDN*) or its long form equivalent (for example, `--bindDN` *bindDN*).

Command Options

`-a, --authzID` ***authzID***
Specify an authorization ID for the user whose password is to be changed. The authorization ID can be in the form `dn:` followed by the DN of the target user, or `u:` followed by the user name of the target user. If this option is not provided, then no authorization ID will be included in the request and the password for the authenticated user will be changed. Do not use this option with the `--provideDNForAuthzID` option.

`-A, --provideDNForAuthzID`
Indicate that the bind DN should be used as the authorization ID for the password modify operation. Do not use this option with the `--authzID` option.

`-c, --currentPassword` ***currentPassword***
Specify the current password for the user. Do not use with `--currentPasswordFile`. The user's current password must be provided in cases in which no authentication is performed, for example, if a user is trying to change his password after it has already expired. The password might also be required by the server based on the password policy configuration even if a bind password was provided.

`-C, --currentPasswordFile` ***currentPasswordFile***
Read the current password from the specified file. Do not use with `--currentPassword`. The user's current password must be provided in cases in which no authentication is performed, for example, if a user is trying to change his password after it has already expired. The password might also be required by the server based on the password policy configuration even if a bind password was provided.

`-J, --control` ***controloid*[:*criticality*[:*value*|::*b64value*|:<*fileurl*]]**
Perform a search with the specified control in search requests sent to the directory server. This option makes it possible to include arbitrary request controls that the client cannot directly support. The value for this option must be in the form:
oid[:*criticality*[:*value*|::*b64value*|:<*fileurl*]]
The elements of this value include:

- **oid.** Use the OID for the control. For certain types of controls, a text name may be used instead of the numeric OID (for search operations, this includes `managedsait` for the manage DSA IT control). This element is required. Human-readable names can be used in place of the OID to reference controls that do not require values using the -J or `control` option. These OID names are the following:

  `accountusable` or `accountusability` — Use in place of the Account Usability Request Control OID: 1.3.6.1.4.1.42.2.27.9.5.8 (no value).

`authzid` or `authorizationidentity` — Use in place of the Authorization Identity Request Control OID: 2.16.840.1.113730.3.4.16 (no value).

`effectiverights` — Use in place of the Get Effective Rights Control OID: 1.3.6.1.4.1.42.2.27.9.5.2 (value = authorization ID).

`managedsait` — Use in place of the Manage DSA IT Control OID: 2.16.840.1.113730.3.4.2 (no value).

`noop` or `no-op` — Use in place of the LDAP No-op Control OID: 1.3.6.1.4.1.4203.1.10.2 (no value).

`pwpolicy` or `password policy` — Use in place of the Password Policy Request Control OID: 1.3.6.1.4.1.42.2.27.8.5.1 (no value).

`subtreedelete` or `treedelete` — Use in place of the Subtree Delete Request Control OID: 1.2.840.113556.1.4.805 (no value).

- **criticality.** If `true`, the control should be marked critical (meaning that the directory server should not process the operation unless it can meet the requirements of this control). If `false`, the control should not be marked critical. If this subcommand is not provided, then the control is not marked critical.
- **value.** Specifies the value for the control. Use this form only if the value can be expressed as a string. Do not use with either the `::`*b64value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **b64value.** Specifies the value for the control in base64-encoded form. Do not use this subcommand with either the `:`*value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **fileurl.** Specifies a URL that references a file from which the value of the control should be taken. Do not use with either the `:`*value* or `::`*b64value* forms. If none of these subcommands is present, then the control will not have a value.

For example, the value `1.3.6.1.4.1.42.2.27.9.5.2:true:dn:uid=dmiller,ou=people,dc=example,dc=com` will include a critical control with an OID of `1.3.6.1.4.1.42.2.27.9.5.2`, marked as critical (true), and with a string value for the authorization ID `dn:uid=dmiller,ou=people,dc=example,dc=com`. Or, you can use the OID names: `effectiverights:true:dn:uid=dmiller,ou=people,dc=example,dc=com`.

`-n, --newPassword` ***newPassword***
Specify the new password that should be assigned to the target user. Do not use this option with `--newPasswordFile`. If neither of these options is provided, then the server will automatically generate a new password for the user if a password generator is configured in the user's password policy.

`-N, --newPasswordFile` ***newPasswordFile***
Read the new password from the specified file that should be assigned to the target user. Do not use this option with `--newPassword`. If neither of these options is provided, then the server automatically generates a new password for the user, if a password generator is configured in the user's password policy.

LDAP Connection Options

`--certNickname` ***nickname***
Use the certificate for certificate-based client authentication.

`-D, --bindDN` *bindDN*
Use the DN when binding to the directory server through simple authentication. If this option is not provided, then the `--authzID` option must be used to specify the authorization ID for the target user, and either the `--currentPassword` or `--currentPasswordFile` option must be provided to specify the current password for the user. (This mode of use will be required for users to change their passwords after the passwords have expired.)

`-h, --hostname` *address*
Contact the directory server on the specified host name or IP address. If it is not provided, then a default address of `localhost` will be used.

`-j, --bindPasswordFile` *bindPasswordFile*
Use the bind password in the specified file when authenticating to the directory server. The option is used for simple authentication, as well as for password-based SASL mechanisms such as CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if no authentication is to be performed. Do not use this option with `--bindPassword`.
SASL is not supported for a proxy server instance.

`-K, --keyStorePath` *keyStorePath*
Use the client keystore certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option should only be necessary if the client needs to present a certificate to the directory server, for example, when using SASL EXTERNAL authentication.
SASL is not supported for a proxy server instance.

`-o, --saslOption` *name=value*
Use the specified option when performing SASL authentication. Multiple SASL options can be provided by using this option multiple times, once for each option. See Using SASL Authentication for more information.

`-p, --port` *port*
Contact the directory server at the specified port. If this option is not provided, then a default port of `389` will be used.

`-P, --trustStorePath` *trustStorePath*
Use the client trust store certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

`-q, --useStartTLS`
Use the StartTLS extended operation when communicating with the directory server. Do not use this option with `--useSSL`.

`--trustStorePassword` *trustStorePassword*
Use the password needed to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (which most trust stores do not require). Do not use this option with `--trustStorePasswordFile`.

`-u, --keyStorePasswordFile` *keyStorePasswordFile*
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePassword`.

-U, --trustStorePasswordFile *trustStorePasswordFile*
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this). Do not use this option with `--trustStorePassword`.

-w, --bindPassword *bindPassword*
Use the bind password when authenticating to the directory server. This option can be used for simple authentication as well as password-based SASL mechanisms. Do not use this option with `--bindPasswordFile`. To prompt for the password, type `-w -`.
SASL is not supported for a proxy server instance.

-W, --keyStorePassword *keyStorePassword*
Use the password needed to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePasswordFile`.

-X, --trustAll
Trust any certificate that the directory server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

-Z, --useSSL
Use the Secure Sockets Layer when communicating with the directory server. If SSL is to be used, then the `--port` option should be used to specify the server's secure port.

Command Input/Output Options

--noPropertiesFile
Indicate that a properties file will not be used to get the default command-line options.

--propertiesFilePath *propertiesFilePath*
Specify the path to the properties file that contains the default command-line options.

General Options

-?, -H, --help
Display command-line usage information for the command and exit without making any attempt to run the command.

-V, --version
Display the version information for the directory server.

Examples

The following examples show how to use the `ldappasswordmodify` command.

Modifying Your User Password

The following command connects to the host (`-h`) using port `1389` (`-p`), specifies the authorization ID `uid=abergin` (`-a`) of an administrator, specifies the user's current password file (`-C`), and changes it with a new one specified in a new password file (`-N`). For Windows platforms, use the file paths where your current and new passwords exist, respectively. For example, use `-C \temp\currentPasswordFile` and `-N \temp\newPasswordFile`.

```
$ ldappasswordmodify -h hostname -p 1389 \
-a "dn:uid=abergin,ou=People,dc=example,dc=com" \
-C /tmp/currentPasswordFile -N /tmp/newPasswordFile
```

```
The LDAP password modify operation was successful
```

Modifying and Generating a Password for Another User

The following command connects to the host (`-h`) using port `1389` (`-p`), specifies the bind DN (`-D`), specifies the bind password file (`-j`), and modifies and generates a password for another user (`-a`) connecting over simple authentication. For Windows platforms, specify the file where the bind password file resides, for example, `-j \temp\bindPasswordFile`.

```
$ ldappasswordmodify -h hostname -p 1389 \
-D "cn=Directory Manager" -j /tmp/bindPasswordFile \
-a "dn:uid=abergin,ou=People,dc=example,dc=com"

The LDAP password modify operation was successful
Generated Password:  blb44hjm
```

Modifying a Password for Another User

The following command connects to the host (`-h`) using port `1389` (`-p`), specifies the bind DN (`-D`), specifies the bind password file (`-j`), and modifies the password with a new one (`-N`) for another user (`-a`) connecting over simple authentication. For Windows platforms, specify the bind password file (for example, `-j \temp\bindPasswordFile`) and the new password file (for example, `-N \temp\newPassword`).

```
$ ldappasswordmodify -h hostname -p 1389 \
-D "cn=Directory Manager" -j /tmp/bindPasswordFile \
-a "dn:uid=abergin,ou=People,dc=example,dc=com" -N /tmp/newPassword

The LDAP password modify operation was successful
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

How to Use a CLI Properties File

The directory server supports the use of a properties file that passes in any default option values used with the `ldappasswordmodify` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. See Using a Properties File With Server Commands for more information.

The following options can be stored in a properties file:

- `authzID`
- `bindDN`
- `bindPassword`
- `bindPasswordFile`
- `currentPassword`
- `currentPasswordFile`
- `control`
- `hostname`
- `keyStorePassword`
- `keyStorePasswordFile`

- keyStorePath
- newPassword
- newPasswordFile
- port
- provideDNForAuthzID
- trustAll
- trustStorePassword
- trustStorePasswordFile
- trustStorePath
- useSSL
- useStartTLS

Entries in the properties file have the following format:

```
toolname.propertyname=propertyvalue
```

For example:

```
ldappasswordmodify.ldapport=12345
```

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldappasswordmodify`
- Windows: *INSTANCE_DIR*`\OUD\bat\ldappasswordmodify.bat`

Related Commands

- ldapcompare
- ldapdelete
- ldapmodify
- ldapsearch

A.1.4.5 ldapsearch

The `ldapsearch` command searches directory server entries.

Synopsis

`ldapsearch` [*options*] [*filter*] [*attributes*]

Description

The `ldapsearch` command can be used to enter a search request to the directory server. The command opens a connection to the directory server, binds to it, and returns all entries that meet the search filter and scope requirements starting from the specified base DN. It can also be used to test other components of the directory server, such as authentication, control, and secure communication mechanisms.

If the `--filename` option is used to specify a file containing one or more search filters, then the search filter should not be included as an option. All trailing options will be interpreted as requested attributes.

If an entry has non-ASCII characters for its name and attributes, such as *sn, givenName, uid,* and *title,* the non-ASCII characters returned by running the `ldapsearch` command are suppressed while printing. You must run the `base64` command to decode the Base64-encoded string.

If no specific attributes are requested, then all user attributes (that is, all non-operational attributes) are returned. If one or more attribute names are listed, then only those attributes are included in the entries that are returned.

Before You Begin

Many UNIX and Linux operating systems provide an installed version of common LDAP client commands, such as `ldapsearch`, `ldapmodify`, and `ldapdelete` in the `/usr/bin` directory. You can check if a version is on your system by entering the command: `which ldapsearch`. If the command returns a value (seen below), you will need to update your `$PATH` to directory server installation directory or create an alias to the directory server instance.

```
$ which ldapsearch (Unix/Linux)
/usr/bin/ldapsearch
```

Options

The `ldapsearch` command accepts an option in either its short form (for example, `-b` *baseDN*) or its long form equivalent (for example, `--baseDN` *baseDN*).

Command Options

`-a, --dereferencePolicy dereferencePolicy`
Specify the dereference alias policy during a search. *Dereference alias* allows you to set an entry to point to another object. If this option is not provided, then a default of `never` will be used. Possible values are the following:

- `always` — Dereference aliases both when finding the base DN and when searching below it.
- `find` — Dereference alias when finding the base DN.
- `never` — Never dereference aliases (default).
- `search` — Dereference aliases when searching below the base DN but not when finding the base DN.

`-A, --typesOnly`
Perform a search to include attribute names in matching entries but not the attribute values. If this option is not provided, then both attribute names and values will be included in the matching entries.

`--assertionFilter filter`
Perform a search using the LDAP assertion control (as defined in RFC 4528 (`http://www.ietf.org/rfc/rfc4528.txt`)) to indicate that the operation should only be processed if the assertion contained in the provided filter is true.

`-b, --baseDN baseDN`
Specify the base DN to use for the search operation. If a file containing multiple filters is provided using the `--filename` option, then this base DN will be used for all of the searches. This is a required option. If a base DN with a null value (`""`) is specified, the server returns the root DSE entry.

-c, --continueOnError
Continue processing even if an error occurs. Use this option when you use multiple search filters in a file (`--filename`). If an error occurs during processing, the server will continue processing the next search filter. Otherwise the command will exit before all searches have been completed.

-C, --persistentSearch ps[:changetype[:changesonly[:entrychangecontrols]]]
Use the persistent search control in the search request to obtain information about changes that are made to entries that match the provided search criteria. The value for this option must be in the form:
`ps[:changetype[:changesonly [:entrychangecontrols]]]`
The elements of this value include:

- `ps` — Required operator.
- `changetype` — Indicates the types of changes for which the client wants to receive notification. It can be any of `add`, `del`, `mod`, or `moddn`, or it can be `all` to register for all change types, or it can be a comma-separated list to register for multiple specific change types. If this element is not provided, then it will default to including `all` change types.
- `changesonly` — If `true`, the client is only notified of changes that occur to matching entries after the search is registered. If `false`, the directory server sends all existing entries in the directory server that match the provided search criteria. If this element is not provided, then it will default to only returning entries for updates that occurred since the search was registered.
- `entrychangecontrols` — If `true`, the directory server includes the entry change notification control in entries sent to the client as a result of changes. If `false`, the entry change notification control is not included. If this element is not provided, then it will default to including the entry change notification controls.

For example, the value `ps:add,del:true:true` returns only entries matching the search criteria that have been added or deleted since the time that the persistent search was registered, and those entries will include entry change notification controls.

--countEntries
Display the total number of matching entries returned by the directory server. If the `--filename` option is used to specify the path to a file containing multiple search filters, the total number of matching entries for all searches is displayed.

-e, --getEffectiveRightsAttribute attribute
Return the effective rights on the specified attribute. This option can be used to specify attributes that would not normally appear in the search results for the entry. For example, use this option to determine if a user has permission to add an attribute that does not currently exist in the entry. The `-e` option requires the `--getEffectiveRightsAuthzid` or `-g` option.

-f, --filename filename
Specify the path to a file that contains one or more filters to use when processing the search operation. If the file contains multiple filters, the file should be structured with one filter per line. The searches will be performed using the same connection to the directory server in the order that they appear in the filter file. If this option is used, any trailing options will be treated as separate attributes. Otherwise the first trailing option must be the search filter.

-g, --getEffectiveRightsAuthzid authzid
Display the effective rights of the user binding with the given *authzid*. This option can be used with the `-e` option but cannot be used with the `-J` option.

-G, --virtualListView before:after:index:count|before:after:value
Retrieve the virtual list view displaying a portion of the total search results. Use one of two patterns to specify the size of the virtual list view:

- `before:after:index:count` — Return the target entry and the specified number of entries *before* the target entry and *after* the target entry. The target entry depends on the `index` and the `count` options. The `count` option can take the following values:

  count=0. The target entry is the entry at the specified *index* position, starting from 1 and relative to the entire list of sorted results.

  count=1. The target entry is the first entry in the list of sorted results.

  count>1. The target entry is the first entry in the portion of the list represented by the fraction *index*/*count*. To target the last result in the list, use an *index* option greater than the *count* option.

  For example, `-G 5:10:2:4` specifies the *index* closest to the beginning of the second quarter of the entire list. If the search yielded 100 entries, the target index would be 26, and this pattern would return entries 21 through 36.

- `before:after:value` — Return the target entry and specified number of entries before and after the target entry. The target entry is the first entry in the sorted results whose sort attribute is greater than or equal to the specified value.

  For example, `-G 5:10:johnson -S sn` returns 16 entries in alphabetical order from the surname attribute: `5` less than `johnson`, the entry equal to or following `johnson`, and the `10` entries after `johnson`.

-J, --control controloid[:criticality[:value|::b64value |:<filePath]]
Perform a search with the specified control in search requests sent to the directory server. This option makes it possible to include arbitrary request controls that the client cannot directly support. The value for this option must be in the form:
`oid[:criticality[:value|::b64value|:<filePath]]`
The elements of this value include:

- **oid.** Use the OID for the control. For certain types of controls, a text name may be used instead of the numeric OID (for search operations, this includes `managedsait` for the manage DSA IT control). This element is required. Human-readable names can be used in place of the OID to reference controls that do not require values using the -J or `control` option. These OID names are the following:

  `accountusable` or `accountusability` — Use in place of the Account Usability Request Control OID: 1.3.6.1.4.1.42.2.27.9.5.8 (no value).

  `authzid` or `authorizationidentity` — Use in place of the Authorization Identity Request Control OID: 2.16.840.1.113730.3.4.16 (no value).

  `effectiverights` — Use in place of the Get Effective Rights Control OID: 1.3.6.1.4.1.42.2.27.9.5.2 (value = authorization ID).

  `managedsait` — Use in place of the Manage DSA IT Control OID: 2.16.840.1.113730.3.4.2 (no value).

  `noop` or `no-op` — Use in place of the LDAP No-op Control OID: 1.3.6.1.4.1.4203.1.10.2 (no value).

  `pwpolicy` or `password policy` — Use in place of the Password Policy Request Control OID: 1.3.6.1.4.1.42.2.27.8.5.1 (no value).

  `subtreedelete` or `treedelete` — Use in place of the Subtree Delete Request Control OID: 1.2.840.113556.1.4.805 (no value).

- **criticality.** If `true`, the control should be marked critical (meaning that the directory server should not process the operation unless it can meet the requirements of this control). If `false`, the control should not be marked critical. If this subcommand is not provided, then the control is not marked critical.
- **value.** Specifies the value for the control. Use this form only if the value can be expressed as a string. Do not use with either the `::`*b64value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **b64value.** Specifies the value for the control in base64-encoded form. Do not use this subcommand with either the `:`*value* or `:<` *fileurl* forms. If none of these subcommands is present, then the control will not have a value.
- **fileurl.** Specifies a URL that references a file from which the value of the control should be taken. Do not use with either the `:`*value* or `::`*b64value* forms. If none of these subcommands is present, then the control will not have a value.

For example, the value `1.3.6.1.4.1.42.2.27.9.5.2:true:dn:uid=dmiller,ou=people,dc=example,dc=com` will include a critical control with an OID of `1.3.6.1.4.1.42.2.27.9.5.2`, marked as critical (true), and with a string value for the authorization ID `dn:uid=dmiller,ou=people,dc=example,dc=com`. Or, you can use the OID names: `effectiverights:true:dn:uid=dmiller,ou=people,dc=example,dc=com`.

`-l, --timeLimit numSeconds`
Set the maximum length of time, in seconds, that the directory server should spend processing any search request. If this option is not provided, no time limit is requested by the client.

> **Note:**
>
> The directory server can enforce a lower time limit than the one that is requested by the client.

`--matchedValuesFilter filter`
Use the LDAP matched values control (as defined in RFC 3876 (`http://www.ietf.org/rfc/rfc3876.txt`)) to indicate that only attribute values matching the specified filter should be included in the search results. This option can be provided multiple times to specify multiple matched values filters.

`-n, --dry-run`
Run in `no-op` mode. That is, report what should happen but do not actually perform any searches or communicate with the server in any way.

`-s, --searchScope scope`
Set the scope for the search operation. The scope value must be one of the following:

- `base` — Search only the entry specified by the `--baseDN` or `-b` option.
- `one` — Search only the entry specified by the `--baseDN` or `-b` option and its immediate children.
- `sub` or `subordinate` — Search the subtree whose base is the entry specified by the `--baseDN` or `-b` option. This is the default option when the `--searchScope` is not provided.

-S, --sortOrder sortOrder
Sort the results before returning them to the client. The sort order is a comma-delimited list of sort keys, where each sort key consists of the following elements:

- `+/- (plus or minus sign)` — Indicates that the sort should be in ascending (`+`) or descending (`-`) order. If this element is omitted, then the sort will be in ascending order.
- `attribute name` — The name of the attribute to use when sorting the data. This element must always be provided.
- `name` or `OID Matching Rule` — An optional colon followed by the name or OID of the matching rule to use to perform the sort. If this element is not provided, then the default ordering matching rule for the specified attribute type will be used. For example, the sort order string `sn,givenName` sorts entries in ascending order first by `sn` and then by `givenName`. Alternately, the value `--modifyTimestamp` will cause the results to be sorted with the most recent values first.

--simplePageSize numEntries
Use the Simple Paged Results control with the given page size.

--subEntries
Use the subentries control to specify that subentries are visible, and normal entries are not.

-Y, --proxyAsauthzID
Use the Proxied Authorization Control to specify the identity of the user for whom the operations should be performed. This will use version 2 of the Proxied Authorization Control as defined in RFC 4370 (`http://www.ietf.org/rfc/rfc4370.txt`). The value of the option should be an authorization ID in the form `dn:` followed by the DN of the target user (for example, `dn:uid=john.doe,ou=People,dc=example,dc=com`), or `u:` followed by the user name (for example, `u:john.doe`). If this option is not provided, proxied authorization is not used.

-z, --sizeLimit numEntries
Set the maximum number of matching entries that the directory server should return to the client. If this option is not provided, then there will be no maximum requested by the client.

> **Note:**
>
> The directory server can enforce a lower size limit than the one that is requested by the client.

LDAP Connection Options

-D, --bindDN bindDN
Use the bind DN to authenticate to the directory server. This option is used when performing simple authentication. The default value for this option is `cn=Directory Manager`. It is not required when using SASL authentication or if no authentication is to be performed.

-E, --reportAuthzID
Use the authorization identity request control (as defined in RFC 3829 (`http://www.ietf.org/rfc/rfc3829.txt`)) in the bind request so that the directory server returns the corresponding authorization ID to the client when authentication has completed. (The line containing the authorization ID will be prefixed with a `#` character, making it a comment if the output is to be interpreted as an LDIF.)

-h, --hostname address
Contact the directory server on the specified host name or IP address. If it is not provided, then a default address of `localhost` will be used.

-j, --bindPasswordFile bindPasswordFile
Use the bind password in the specified file when authenticating to the directory server. The option is used for simple authentication, as well as for password-based SASL mechanisms such as CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if no authentication is to be performed. Do not use this option with `--bindPassword`.
SASL is not supported for a proxy server instance.

-K, --keyStorePath keyStorePath
Use the client keystore certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option should only be necessary if the client needs to present a certificate to the directory server, for example, when using SASL EXTERNAL authentication.
SASL is not supported for a proxy server instance.

-N, --certNickName certNickName
Use the specified certificate for certificate-based client authentication.

-o, --saslOption name=value
Use the specified option when performing SASL authentication. Multiple SASL options can be provided by using this option multiple times, once for each option. See Configuring SASL Authentication for more information on using SASL authentication in clients.
SASL is not supported for a proxy server instance.

-p, --port port
Contact the directory server at the specified port. If this option is not provided, then a default port of `389` will be used.

-P, --trustStorePath trustStorePath
Use the client trust store certificate in the specified path for secure communication when using the SSL or the StartTLS extended operation. This option is not needed if `--trustAll` is used, although a trust store should be used when working in a production environment.

-q, --useStartTLS
Use the StartTLS Extended Operation extended operation when communicating with the directory server. Do not use this option with `--useSSL`.

-r, --useSASLExternal
Use the SASL EXTERNAL mechanism for authentication, which attempts to identify the client by using an SSL certificate that it presents to the directory server. If you use this option, then you must also provide the `--keyStorePath` option to specify the path to the client keystore and you must use either the `--useSSL` or the `--useStartTLS` option to establish a secure communication channel with the server.
SASL is not supported for a proxy server instance.

--trustStorePassword trustStorePassword
Use the password needed to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (which most trust stores do not require). Do not use this option with `--trustStorePasswordFile`.

-u, --keyStorePasswordFile keyStorePasswordFile
Use the password in the specified file to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePassword`.

--usePasswordPolicyControl
Use the Password Policy Request Control in the bind request so that the directory server returns the corresponding result control in the bind response. Use this option to obtain information about any warnings or errors regarding the state of the client's account.

-U, --trustStorePasswordFile trustStorePasswordFile
Use the password in the specified file to access the certificates in the client trust store. This option is only required if `--trustStorePath` is used and the specified trust store requires a password to access its contents (most trust stores do not require this). Do not use this option with `--trustStorePassword`.

-V, --ldapVersion version
Set the LDAP protocol version that the client should use when communicating with the directory server. The value must be either `2` (for LDAPv2 communication) or `3` (for LDAPv3). If this option is not provided, then the client will use LDAPv3.

-w, --bindPassword bindPassword
Use the bind password when authenticating to the directory server. This option can be used for simple authentication as well as password-based SASL mechanisms. Do not use this option with `--bindPasswordFile`. To prompt for the password, type `-w -`.
SASL is not supported for a proxy server instance.

-W, --keyStorePassword keyStorePassword
Use the password needed to access the certificates in the client keystore. This option is only required if `--keyStorePath` is used. Do not use this option with `--keyStorePasswordFile`.

-X, --trustAll
Trust any certificate that the directory server might present during SSL or StartTLS negotiation. This option can be used for convenience and testing purposes, but for security reasons a trust store should be used to determine whether the client should accept the server certificate.

-Z, --useSSL
Use SSL when communicating with the directory server. If SSL is to be used, then use the `--port` option to specify the server's secure port.

Command Input/Output Options

--noPropertiesFile
Indicate that a properties file will not be used to get the default command-line options.

--propertiesFilePath propertiesFilePath
Specify the path to the properties file that contains the default command-line options.

-T, --dontWrap
Do not wrap long lines when displaying matching entries. If this option is not provided, then long lines will be wrapped (in a manner compatible with the LDIF specification) to fit on an 80-column terminal.

-v, --verbose
Run in verbose mode, displaying process and diagnostic information on standard output.

General Options

`-?, -H, --help`
Display command-line usage information for the command and exit without making any attempt to run the command.

`-V, --version`
Display the version information for the directory server.

Examples

The following examples show how to use the `ldapsearch` command. For additional examples, see About Searching Directory Data.

Returning All Entries

The following command returns all entries on the directory server. The command connects to the default port `1389` (`-p`) on the host (`-h`), specifies the base DN as `example.com` (`-b`), and returns all entries by using the search filter `(objectclass=*)`. Because the scope (`-s`) is not specified, the scope is set to the default value of `sub`, the full subtree of the base DN. Because no attributes are specified, the command returns all attributes and values.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(objectclass=*)"

dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example

dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups

dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniquenames
objectClass: top
ou: Groups
cn: Directory Administrators
uniquemember: uid=kvaughan, ou=People, dc=example,dc=com
uniquemember: uid=rdaugherty, ou=People, dc=example,dc=com
uniquemember: uid=hmiller, ou=People, dc=example,dc=com
```

Returning Attribute Names but No Values

The following command returns the attribute names (`-A`) but no values. The command connects to the default port `1389` (`-p`) on the host (`-h`), specifies the base DN as `dc=example,dc=com` (`-b`), matches all entries by using the search filter `objectclass=*`, and returns three (`-z`) entries. Using the `-A` option is a convenient way to check if an attribute is present in the database.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com -A -z 3 "(objectclass=*)"

dn: dc=example,dc=com
objectClass
dc

dn: ou=Groups,dc=example,dc=com
objectClass
ou
```

```
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass
ou
cn
uniquemember
```

Returning Specific Attribute Values

The following command returns a specific attribute and its value. The command connects to the port `1389` (`-p`) on the host (`-h`), specifies the base DN as `dc=example,dc=com` (`-b`), matches all entries by using the search filter `cn=Sam Carter`, and returns the value of the attribute, `telephonenumber`.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(cn=Sam Carter)" telephoneNumber

dn: uid=scarter,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 4798
```

Returning the Root DSE

The root DSE is a special entry that provides information about the directory server's name, version, naming contexts, and supported features. You specify the root DSE by using a base DN with a null value (for example, `-b ""`) from which the directory server searches below all public naming contexts by default. You can override the null base DN default by specifying specific sets of base DNs with the `subordinate-base-dn` property by using the `dsconfig` command. The following example connects to the default port `1389` (`-p`) on the host (`-h`), specifies the root DSE as an empty base entry (`-b`), specifies the scope of the search to `base` (`-s`), matches all entries by using the search filter `objectclass=*`, and returns the directory server's root DSE information for supported controls:

```
$ ldapsearch -h hostname -p 1389 -b "" -s base "(objectclass=*)" supportedControl

dn:
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.840.113556.1.4.473
supportedControl: 1.2.840.113556.1.4.805
...
```

Searching by Using Server Authentication

Consider the following example of a command that uses the SSL option to run a search with server authentication. The command specifies the host name (`-h`), SSL port `1636` (`-p`), base DN (`-b`), the bind DN (`-D`), the bind password (`-w`), trust store file path (`-P`), and the entity's given name. For Windows platforms, specify the paths for trust store file (for example, `-P \certs\cert.db`).

```
$ ldapsearch -h hostname -p 1636 -b "dc=example,dc=com" \
-D "uid=scarter,ou=people,dc=example,dc=com" -w bindPassword \
-P /home/scarter/certs/cert.db "(givenname=Sam)"
```

Searching by Using Client Authentication

The following command uses the SSL option to perform a search by using client authentication. The command uses SSL (`-Z`) with the SSL port (`-p`) and specifies the trust store file path (`-P`), the certificate nickname (`-N`), the keystore file path (`-K`), the keystore password (`-W`) and the entity's given name `(givenname=Sam)`. For Windows platforms, specify the paths for the trust store file (for example, `-P \certs\cert.db`), and the keystore file (for example, `-K \security\key.db`):

```
$ ldapsearch -h hostname -p 1636 -b "dc=example,dc=com" \
-Z -P /home/scarter/security/cert.db -N "sccert" \
-K /home/scarter/security/key.db -W KeyPassword \
"(givenname=Sam)"
```

Returning the Effective Rights of a User

The following command returns the effective rights granted to a user, in addition to the user's attribute entries. Only a directory administrator can access this information for another user. The command specifies the host name (`-h`), port `1389` (`-p`), bindDN (`-D`), bindDN password (`-w`), base DN (`-b`), control spec option that includes the OID name `effectiverights` (alternately, you can enter the OID equivalent: `1.3.6.1.4.1.42.2.27.9.5.2`), search filter `objectclass=*`, and the `aclRights` attribute.

```
$ ldapsearch -h hostname -p 1389 -D "cn=Directory Manager" -j pwd-file \
-b dc=example,dc=com -J "1.3.6.1.4.1.42.2.27.9.5.2"  "(objectclass=*)" \
aclRights

dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

Returning the Schema

The following command searches the `cn=schema` entry for the object classes and attributes defined on the directory instance. The command connects to the port `1389` (`-p`) on the host (`-h`), sets the scope of the search to `base` (`-s`), matches all entries by using the search filter `(objectclass=\*)` and returns the objectClass definitions in the schema entry, `cn=schema`. You can also use the `+` symbol to view the schema. Place it after the search filter.

```
$ ldapsearch -h hostname -p 1389 -b cn=schema -s base "(objectclass=*)" objectClasses

dn: cn=schema
objectClasses: ( 2.5.6.0 NAME 'top' ABSTRACT MUST objectClass X-ORIGIN 'RFC 4512
' )
objectClasses: ( 2.5.6.1 NAME 'alias' SUP top STRUCTURAL MUST aliasedObjectName
X-ORIGIN 'RFC 4512' )
objectClasses: ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL MUST c MAY ( searchGu
ide $ description ) X-ORIGIN 'RFC 4519' )
objectClasses: ( 2.5.6.3 NAME 'locality' SUP top STRUCTURAL MAY ( street $ seeAl
so $ searchGuide $ st $ l $ description ) X-ORIGIN 'RFC 4519' )
...
```

Performing a Persistent Search

The `ldapsearch` command provides an option to run a persistent search (`-C`) that keeps the connection open and displays the entries that matching the scope and filter whenever any changes (`add`, `delete`, `mod`, or `all`) occur. The command connects to the port `1389` (`-p`), sets the scope of the search to `base` (`-s`), and matches all entries by using the search filter `(objectclass=\*)`. You can quit out of the search by pressing `Control-C`.

```
$ ldapsearch -b dc=example,dc=com -p 1389 -D "cn=Directory Manager" \
-j pwd-file -C ps:add:true:true "(objectclass=*)"
```

Viewing ACI Attributes

The following command displays the access control instruction (ACI) attributes from the specified base DN. The command connects to the port `1389` (`-p`), sets the scope of the search to `base` (`-s`), matches all entries using the search filter `(objectclass=\*)` and specifies the `aci` attribute.

```
$ ldapsearch -p 1389 -D "cn=Directory Manager" -j pwd-file -b dc=example,dc=com \
-s base "(objectclass=*)" aci

dn: dc=example,dc=com
aci: (target ="ldap:///dc=example,dc=com")(targetattr h3.="userPassword")(version
3.0;acl "Anonymous read-search access";allow (read, search, compare)(userdn = "
ldap:///anyone");)
aci: (target="ldap:///dc=example,dc=com") (targetattr = "*")(version 3.0; acl "a
llow all Admin group"; allow(all) groupdn = "ldap:///cn=Directory Administrator
s,ou=Groups,dc=example,dc=com";)
```

Viewing Monitoring Information

The following command searches the cn=monitor entry for information on the activity on the directory server. The command specifies the host name (`-h`), port (`-p`), base DN (`-b`) for `cn=monitor`, authenticates using the bind DN (`-D`) and bind password (`-w`) and specifies the filter `(objectclass=\*)`.

```
$ ldapsearch --useSSL -X -h hostname -p 4444 -b cn=monitor -D "cn=Directory Manager" \
-j pwd-file "(objectclass=*)"

dn: cn=monitor
objectClass: top
objectClass: extensibleObject
objectClass: ds-monitor-entry
currentTime: 20070803161832Z
startTime: 20070803132044Z
productName: Oracle Unified Directory
...
```

Searching by Using a Properties File

The directory server supports the use of a *properties file* that passes in any default option values used with the `ldapsearch` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. See Using a Properties File With Server Commands for more information.

The following options can be stored in a properties file:

- `assertionFilter`
- `bindDN`
- `bindPassword`
- `bindPasswordFile`

- certNickname
- continueOnError
- control
- countEntries
- dereferencePolicy
- dry-run
- dontWrap
- filename
- getEffectiveRightsAttribute
- getEffectiveRightsAuthzid
- hostname
- keyStorePassword
- keyStorePasswordFile
- keyStorePath
- ldapVersion
- matchedValuesFilter
- persistentSearch
- port
- proxyAs
- reportAuthzID
- saslOption

  SASL is not supported for a proxy server instance.
- searchScope
- simplePageSize
- sizeLimit
- sortOrder
- timeLimit
- trustAll
- trustStorePassword
- trustStorePasswordFile
- trustStorePath
- typesOnly
- usePasswordPolicyControl
- useSASLExternal

  SASL is not supported for a proxy server instance.
- useSSL

- useStartTLS
- verbose
- virtualListView

To Search by Using a Properties File

1. Create a properties file in any text editor. Here, save the file as `tools.properties`.

```
hostname=host
port=1389
bindDN=cn=Directory Manager
bindPassword=password
baseDN=dc=example,dc=com
searchScope=sub
sortOrder=givenName
virtualListView=0:2:1:0
```

2. Use `ldapsearch` with the `--propertiesFilePath` option. `$ldapsearch --propertiesFilePath tools.properties "(objectclass=*)"`

Search Attributes

A number of special search attributes can also be used for various purposes, including the following:

`*`This symbol indicates that all user attributes should be included in the entries returned by the directory server.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(objectclass=*)" *
```

`+`This symbol indicates that all operational attributes are to be included in the entries returned by the directory server. By default, no operational attributes will be returned. However, even if this is specified, there might be some operational attributes that are not returned automatically for some reason, such as if an expensive computation is required to construct the value). On some systems, you might need to escape the `+` symbol by enclosing it in quotation marks, `"+"` or by using a backslash, `\+`.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(objectclass=*)" "+"
```

`1.1`This indicates that no attribute values should be included in the matching entries. On some systems, you might need to escape the `1.1` character by enclosing it in quotation marks, `"1.1"`, or by using a backslash, `\1.1`.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(objectclass=*)" "1.1"
```

`@_objectclass_`This indicates that all attributes associated with the specified object class should be included in the entries returned by the server. For example, `@person` indicates that the server should include all attributes associated with the `person` object class.

```
$ ldapsearch -h hostname -p 1389 -b dc=example,dc=com "(objectclass=*)" @person
```

Exit Codes

An exit code of 0 indicates that the operation completed successfully. A nonzero exit code indicates that an error occurred during processing.

Location

- UNIX and Linux: *INSTANCE_DIR*`/OUD/bin/ldapsearch`

- Windows: *INSTANCE_DIR*`\OUD\bat\ldapsearch.bat`

Related Commands

- ldapcompare
- ldapdelete
- ldapmodify
- ldappasswordmodify

A.2 LDAP Controls and Operations Reference

Oracle Unified Directory provides the reference information on standard LDAP controls and extended operations.

- Supported LDAP Controls
- Supported Extended Operations

For information about using the LDAP controls, see Searching Using Controls.

A.2.1 Supported LDAP Controls

A supported control is a mechanism for identifying the request control supported by the Oracle Unified Directory.

The object identifier of these controls are listed in the `supportedControl` attribute of the server's root DSE.

Table A-4 lists the controls supported by the directory server.

Table A-4 LDAP Controls Supported by the Directory Server

OID	LDAP Control	RFC or draft
1.2.826.0.1.3344810.2.3	Matched Values Control	RFC3876
1.2.840.113556.1.4.319	Page Results Control	RFC2696
1.2.840.113556.1.4.473	Server-side Sort Control	RFC2891
1.2.840.113556.1.4.805	Subtree Delete Control	Draft
1.3.6.1.1.12	Assertion Control	RFC4528
1.3.6.1.1.13.1	LDAP Pre-read Control	RFC4527
1.3.6.1.1.13.2	LDAP Post-read Control	RFC4527
1.3.6.1.4.1.26027.1.5.2	Replication Repair Control	
1.3.6.1.4.1.4203.1.10.2	LDAP No-Op Control	Draft
1.3.6.1.4.1.42.2.27.8.5.1	Password Policy Control	Draft
1.3.6.1.4.1.42.2.27.9.5.2	Get Effective Rights Control	Draft
1.3.6.1.4.1.42.2.27.9.5.8	Account Usability Control	
1.3.6.1.4.1.42.2.27.9.5.9	CSN (Change Number Control)	**Note:** This control is for internal use only.
1.3.6.1.4.1.4203.1.10.1	LDAP Subentry Request Control	RFC3672
1.3.6.1.4.1.26027.2.3.1	Join Search Control	

Table A-4 (Cont.) LDAP Controls Supported by the Directory Server

OID	LDAP Control	RFC or draft
1.3.6.1.4.1.26027.2.3.2	Proximity Search Control	
1.3.6.1.4.1.26027.2.3.4	External Changelog Cookie v2 Control	**Note:** This control is for internal use only.
2.16.840.1.113730.3.4.4	Password Expired Control	Draft
2.16.840.1.113730.3.4.5	Password Expiration Warning Control	Draft
2.16.840.1.113730.3.4.12	Proxy Authorization v1 Control	Draft
2.16.840.1.113730.3.4.18	Proxy Authorization v2 Control	RFC4370
2.16.840.1.113730.3.4.16	Authorization Identity Request Control	RFC3829
2.16.840.1.113730.3.4.17	Real Attributes Only Control	
2.16.840.1.113730.3.4.19	Virtual Attributes Only Control	
2.16.840.1.113730.3.4.2	Manage DSA IT Control	RFC3296
2.16.840.1.113730.3.4.3	Persistent Search Control	Draft
2.16.840.1.113730.3.4.9	Virtual List View Control	Draft
2.16.840.1.113894.1.8.21	OID Search Count Control	**Note:** This control is used to ensure compatibility with Oracle Internet Directory. For more information about the control, see OID Search Count Request Control.

Table A-4 (Cont.) LDAP Controls Supported by the Directory Server

OID	LDAP Control	RFC or draft
2.16.840.1.113894.1.8.31	Execution context ID (ECID)	ECID is an unique identifier used across several Oracle product components to track requests within the same transaction. It is used in OUD to track LDAP requests coming in from the client for a given ECID.

Note: This control is for Oracle internal use

Table A-4 (Cont.) LDAP Controls Supported by the Directory Server

OID	LDAP Control	RFC or draft
		only.

If you have installed a proxy instance, see Table A-5, which lists the controls supported by the proxy as well as by the remote LDAP servers.

Table A-5 LDAP Controls Supported by the Proxy

OID	LDAP Control	RFC or draft	Supported by Proxy Workflow Element	Supported by Distribution Algorithm	Supported by Remote ODSEE	Supported by Remote Oracle Unified Directory Server	Notes
1.2.826.0.1.3344810.2.3	Matched Values Control	RFC3876	Yes	Yes	No	Yes	
1.2.840.113556.1.4.319	Page Results Control	RFC2696	Yes	No	No	Yes	
1.2.840.113556.1.4.473	Server-side Sort Control	RFC2891	Yes	No	Yes	Yes	Supported if all targeted entries are on the same remote LDAP server, and that remote LDAP server supports server-side LDAP control.
1.2.840.113556.1.4.805	Subtree Delete Control	Draft	Yes	No	No	Yes	Supported if all targeted entries are on the same remote LDAP server, and that remote LDAP server supports subtree delete LDAP control. Not supported by the distribution algorithm because targeted entries can span multiple remote LDAP servers.
1.3.6.1.4.1.26027.2.3.2	Proximity Search Control		Yes	Yes	Yes	Yes	

Table A-5 (Cont.) LDAP Controls Supported by the Proxy

OID	LDAP Control	RFC or draft	Supported by Proxy Workflow Element	Supported by Distribution Algorithm	Supported by Remote ODSEE	Supported by Remote Oracle Unified Directory Server	Notes
1.3.6.1.1.12	Assertion Control	RFC4528	Yes	Yes	No	Yes	Supported if the remote LDAP server that hosts the targeted entry also supports assertion control. Therefore not supported in proxy configurations where all remote LDAP servers run Oracle Directory Server Enterprise Edition.
1.3.6.1.1.13.1	LDAP Pre-read Control	RFC4527	Yes	Yes	Complies sufficiently for the proxy to work	Yes	Supported if the remote LDAP servers that host the targeted entries also support LDAP pre-read control. Required for the global index catalog. In Oracle Unified Directory directory servers, this control must be enabled.
1.3.6.1.1.13.2	LDAP Post-read Control	RFC4527	Yes	Yes	No	Yes	Supported if the remote LDAP servers that hosts the targeted entries also support LDAP post-read control. Therefore not supported in proxy configurations where all remote LDAP servers run Oracle Directory Server Enterprise Edition. In Oracle Unified Directory directory servers, this control must be enabled.

Table A-5 (Cont.) LDAP Controls Supported by the Proxy

OID	LDAP Control	RFC or draft	Supported by Proxy Workflow Element	Supported by Distribution Algorithm	Supported by Remote ODSEE	Supported by Remote Oracle Unified Directory Server	Notes
1.3.6.1.4.1.26027.1.5.2	Replication Repair Control		No	No	No	Yes	Not supported by the proxy. To repair data inconsistency across remote LDAP servers, bypass the proxy and send the control directly to the remote LDAP servers running Oracle Unified Directory. For remote LDAP servers running Oracle Directory Server Enterprise Edition, see the `dsrepair` command in the Oracle Directory Server Enterprise Edition documentation.
1.3.6.1.4.1.4203.1.10.2	LDAP No-Op Control	Draft	Yes	Yes	No	Yes	Supported if the remote LDAP servers that host the targeted entries also support the LDAP no-op control. Therefore not supported in proxy configurations where all remote LDAP servers run Oracle Directory Server Enterprise Edition.
1.3.6.1.4.1.42.2.27.8.5.1	Password Policy Control	Draft	Yes	Yes	Yes	Yes	
1.3.6.1.4.1.42.2.27.9.5.2	Get Effective Rights Control	Draft	Yes	Yes	Yes	Yes	If this control is to be used by a configuration of the proxy where remote LDAP servers run Oracle Unified Directory, then the aclRights and aclRightsInfo controls need to be authorized in Oracle Unified Directory, if you have sufficient credentials.

Table A-5 (Cont.) LDAP Controls Supported by the Proxy

OID	LDAP Control	RFC or draft	Supported by Proxy Workflow Element	Supported by Distribution Algorithm	Supported by Remote ODSEE	Supported by Remote Oracle Unified Directory Server	Notes
1.3.6.1.4.1.42.2.27.9.5.8	Account Usability Control		Yes	Yes	Yes	Yes	
1.3.6.1.4.1.4203.1.10.1	LDAP Subentry Request Control	RFC3672	Yes	Yes	No	Yes	Supported if the remote LDAP servers that host the targeted entries also support the LDAP sub-entry control.
1.3.6.1.4.1.26027.1.5.4	External Changelog Cookie Control		Yes	Yes	No	Yes	
1.3.6.1.4.1.42.2.27.9.5.9	CSN (Change Number Control) **Note:** This control is for internal use only.		Yes	Yes	Yes	Yes	Dedicated to replication, appropriate for modifyRequest, delRequest, and modDNRequest LDAP messages. Required for the global index catalog.
2.16.840.1.113730.3.4.12	Proxy Authorization v1 Control	Draft	Yes	Yes	Yes	Yes	Supported if the remote LDAP servers that host the targeted entries also support the proxy-authorization v1 control. If the proxy is configured in this control mode, the remote LDAP server must also support the get effective rights control.
2.16.840.1.113730.3.4.18	Proxy Authorization v2 Control	RFC4370	Yes	Yes	Yes	Yes	Supported if the remote LDAP servers that host the targeted entries also support the proxy-authorization v2 control. If the proxy is configured in this control mode, the remote LDAP server must also support the get effective rights control.

Table A-5 (Cont.) LDAP Controls Supported by the Proxy

OID	LDAP Control	RFC or draft	Supported by Proxy Workflow Element	Supported by Distribution Algorithm	Supported by Remote ODSEE	Supported by Remote Oracle Unified Directory Server	Notes
2.16.840.1.113730.3.4.16	Authorization Identity Request Control	RFC3829	Yes	Yes	Yes	Yes	Supported if the remote LDAP server that hosts the target entry also supports the authorization identity request control.
2.16.840.1.113730.3.4.17	Real Attributes Only Control		Yes	Yes	Yes	Yes	Supported if the remote LDAP servers that host the targeted entries also support the real attributes only control.
2.16.840.1.113730.3.4.19	Virtual Attributes Only Control		Yes	Yes	Yes	Yes	Supported if the remote LDAP servers that host the targeted entries also support the virtual attributes only request control.
2.16.840.1.113730.3.4.2	Manage DSA IT	RFC3296	Yes	Yes	Yes	Yes	
2.16.840.1.113730.3.4.3	Persistent Search Control	Draft	Yes	Yes	Yes	Yes	Supported if the remote LDAP servers that host the targeted entries also support the persistent search control.
2.16.840.1.113730.3.4.9	Virtual List View Control	Draft	Yes	No	Yes	Yes	Supported if all of the targeted entries are located on the same remote LDAP server, and that server supports virtual list view control.

A.2.2 Supported Extended Operations

A supported extension is a mechanism for identifying the extended operation supported by the Oracle Unified Directory.

The object identifier of these extended operations are listed in the `supportedExtension` attribute of the server's root DSE.

Table A-6 lists the extended operations supported by the Oracle Unified Directory.

Table A-6 Extended Operations Supported by the Oracle Unified Directory

OID	Extended Operation
1.3.6.1.1.8	cancel extended operation
1.3.6.1.4.1.1466.20037	StartTLS extended operation
1.3.6.1.4.1.26027.1.6.1	Password Policy State extended operation
1.3.6.1.4.1.26027.1.6.2	Get Connection ID extended operation
1.3.6.1.4.1.26027.1.6.3	Get Symmetric Key extended operation
1.3.6.1.4.1.4203.1.11.1	Password Modify extended operation
1.3.6.1.4.1.4203.1.11.3	"Who Am I?" extended operation

A.3 Standards and Specifications Supported by Oracle Unified Directory

Oracle Unified Directory supports various standards and specifications, such as RFCs, internet drafts, protocols, and cipher suites.

- RFCs Supported by Oracle Unified Directory
- Internet Drafts Supported by Oracle Unified Directory
- Other Specifications Supported by Oracle Unified Directory
- Enabling FIPS Mode on OUD Server
- Supported TLS Protocols and Cipher Suites by Oracle Unified Directory
- Overview of Basic Encoding Rules
- Authenticating Using CRAM-MD5 SASL Mechanism

A.3.1 RFCs Supported by Oracle Unified Directory

Oracle Unified Directory is continuously being updated to ensure that it conforms to the newer protocols.

Table A-7 contains a list of the RFCs currently supported by Oracle Unified Directory.

Table A-7 Supported RFCs

Number	Description
RFC 1274	The COSINE and Internet X.500 Schema
RFC 1321	The MD5 Message-Digest Algorithm
RFC 1777	Lightweight Directory Access Protocol (v2)
RFC 1778	The String Representation of Standard Attribute Syntaxes
RFC 1779	A String Representation of Distinguished Names
RFC 2079	Definition of an X.500 Attribute Type and an Object Class to Hold Uniform Resource Identifiers (URIs)
RFC 2222	Simple Authentication and Security Layer (SASL)
RFC 2247	Using Domains in LDAP/X.500 Distinguished Names

Table A-7 (Cont.) Supported RFCs

Number	Description
RFC 2251	Lightweight Directory Access Protocol (v3)
RFC 2252	Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions
RFC 2254	The String Representation of LDAP Search Filters
RFC 2255	The LDAP URL Format
RFC 2256	A Summary of the X.500(96) User Schema for use with LDAPv3
RFC 2377	Naming Plan for Internet Directory-Enabled Applications
RFC 2605	Directory Server Monitoring MIB
RFC 2649	An LDAP Control and Schema for Holding Operation Signatures
RFC 2696	LDAP Control Extension for Simple Paged Results Manipulation
RFC 2713	Schema for Representing Java(tm) Objects in an LDAP Directory
RFC 2714	Schema for Representing CORBA Object References in an LDAP Directory
RFC 2739	Calendar Attributes for vCard and LDAP
RFC 2788	Network Services Monitoring MIB
RFC 2798	Definition of the inetOrgPerson LDAP Object Class
RFC 2829	Authentication Methods for LDAP
RFC 2830	Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security
RFC 2831	Using Digest Authentication as a SASL Mechanism
RFC 2849	The LDAP Data Interchange Format (LDIF) - Technical Specification
RFC 2891	LDAP Control Extension for Server Side Sorting of Search Results
RFC 2926	Conversion of LDAP Schemas to and from SLP Templates
RFC 3045	Storing Vendor Information in the LDAP root DSE
RFC 3062	LDAP Password Modify Extended Operation
RFC 3112	LDAP Authentication Password Schema
RFC 3174	US Secure Hash Algorithm 1 (SHA1)
RFC 3296	Named Subordinate References in Lightweight Directory Access Protocol (LDAP) Directories
RFC 3377	Lightweight Directory Access Protocol (v3)
RFC 3377	Lightweight Directory Access Protocol (v3): Technical Specification
RFC 3383	Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP)
RFC 3454	Preparation of Internationalized Strings ("stringprep")
RFC 3546	Transport Layer Security (TLS) Extensions
RFC 3671	Collective Attributes in the Lightweight Directory Access Protocol (LDAP)
RFC 3672	Subentries in the Lightweight Directory Access Protocol (LDAP)

Table A-7 (Cont.) Supported RFCs

Number	Description
RFC 3673	Lightweight Directory Access Protocol version 3 (LDAPv3): All Operational Attributes
RFC 3674	Feature Discovery in Lightweight Directory Access Protocol (LDAP)
RFC 3698	Lightweight Directory Access Protocol (LDAP): Additional Matching Rules
RFC 3771	Lightweight Directory Access Protocol (LDAP) Intermediate Response Message
RFC 3829	Lightweight Directory Access Protocol (LDAP) Authorization Identity Request and Response Controls
RFC 3866	Language Tags and Ranges in the Lightweight Directory Access Protocol (LDAP)
RFC 3876	Returning Matched Values with the Lightweight Directory Access Protocol version 3 (LDAPv3)
RFC 3909	Lightweight Directory Access Protocol (LDAP) Cancel Operation
RFC 4370	Lightweight Directory Access Protocol (LDAP) Proxied Authorization Control
RFC 4403	Lightweight Directory Access Protocol (LDAP) Schema for Universal Description, Discovery, and Integration version 3 (UDDIv3)
RFC 4422	Simple Authentication and Security Layer (SASL)
RFC 4505	Anonymous Simple Authentication and Security Layer (SASL) Mechanism
RFC 4510	Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map
RFC 4511	Lightweight Directory Access Protocol (LDAP): The Protocol
RFC 4512	Lightweight Directory Access Protocol (LDAP): Directory Information Models
RFC 4513	Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms
RFC 4514	Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names
RFC 4515	Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters
RFC 4516	Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator
RFC 4517	Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules
RFC 4518	Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation
RFC 4519	Lightweight Directory Access Protocol (LDAP): Schema for User Applications
RFC 4520	Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP)
RFC 4522	Lightweight Directory Access Protocol (LDAP): The Binary Encoding Option

Table A-7 (Cont.) Supported RFCs

Number	Description
RFC 4524	COSINE LDAP/X.500 Schema
RFC 4525	Lightweight Directory Access Protocol (LDAP) Modify-Increment Extension
RFC 4526	Lightweight Directory Access Protocol (LDAP) Absolute True and False Filters
RFC 4527	Lightweight Directory Access Protocol (LDAP) Read Entry Controls
RFC 4528	Lightweight Directory Access Protocol (LDAP) Assertion Control
RFC 4529	Requesting Attributes by Object Class in the Lightweight Directory Access Protocol (LDAP)
RFC 4530	Lightweight Directory Access Protocol (LDAP) entryUUID Operational Attribute
RFC 4532	Lightweight Directory Access Protocol (LDAP) "Who am I?" Operation
RFC 4616	The PLAIN Simple Authentication and Security Layer (SASL) Mechanism
RFC 4634	US Secure Hash Algorithms (SHA and HMAC-SHA)
RFC 4752	The Kerberos V5 ("GSSAPI") SASL Mechanism
RFC 5020	The Lightweight Directory Access Protocol (LDAP) entryDN Operational Attribute
RFC 5246	The Transport Layer Security (TLS) Protocol Version 1.2

A.3.2 Internet Drafts Supported by Oracle Unified Directory

Oracle Unified Directory supports the Internet Engineering Task Force (IETF) and other internet drafts.

Table A-8 contains a list of Internet drafts supported by Oracle Unified Directory.

Table A-8 Internet Drafts Supported by Oracle Unified Directory

Document	Description
draft-armijo-ldap-treedelete	Tree Delete Control
draft-behera-ldap-password-policy	Password Policy for LDAP Directories
draft-furuseth-ldap-untypedobject	Structural object class 'untypedObject' for LDAP/X.500
draft-good-ldap-changelog	Definition of an Object Class to Hold LDAP Change Records
draft-haripriya-dynamicgroup	LDAP: Dynamic Groups for LDAPv3
draft-howard-namedobject	A Structural Object Class for Arbitrary Auxiliary Object Classes
draft-howard-rfc2307bis	An Approach for Using LDAP as a Network Information Service
draft-ietf-boreham-numsubordinates	numSubordinates LDAP Operational Attribute
draft-ietf-ldapext-ldapv3-dupent	LDAP Control for a Duplicate Entry Representation of Search Results
draft-ietf-ldapext-ldapv3-vlv	LDAP Extensions for Scrolling View Browsing of Search Results

Table A-8 (Cont.) Internet Drafts Supported by Oracle Unified Directory

Document	Description
draft-ietf-ldapext-psearch	Persistent Search: A Simple LDAP Change Notification Mechanism
draft-ietf-ldup-subentry	LDAP Subentry Schema
draft-ietf-sasl-crammd5	The CRAM-MD5 SASL Mechanism
draft-ietf-sasl-rfc2831bis	Using Digest Authentication as a SASL Mechanism
draft-poitou-ldap-schema-update	LDAP Schema Update Procedures
draft-sermersheim-ldap-subordinate-scope	Subordinate Subtree Search Scope for LDAP
draft-vchu-ldap-pwd-policy	Password Policy for LDAP Directories
draft-wahl-ldap-adminaddr	LDAP Administrator Address Attribute
draft-weltman-ldapv3-proxy	LDAP Proxied Authorization Control
draft-zeilenga-ldap-noop	The LDAP No-Op Control
draft-zeilenga-ldap-entrydn	The LDAP entryDN Operational Attribute

A.3.3 Other Specifications Supported by Oracle Unified Directory

Oracle Unified Directory supports other standards and documents like the OASIS Directory Services Markup Language v2.0 and Secure Hash Standard.

Table A-9 contains a list of documents and standards supported by Oracle Unified Directory.

Table A-9 Other Specifications Supported by Oracle Unified Directory

Number	Description
DSMLv2.doc	OASIS Directory Services Markup Language v2.0 Documentation
DSMLv2.xsd	OASIS Directory Services Markup Language v2.0 Standard
FIPS 180-1	Secure Hash Standard (SHA-1)
FIPS 180-2	Secure Hash Standard (SHS) (FIPS PUB 180-2)

A.3.4 Enabling FIPS Mode on OUD Server

To enable FIPS mode on OUD server:

> **Note:**
>
> - As a prerequisite, OUD server must be installed and configured with a correct version of JDK.
> - See Enabling FIPS 140-2 Mode From Java Options in Administering Security for Oracle WebLogic Server for detailed steps.

1. Update java security file of the JDK instance referred by your IDM WebLogic domain, as follows:

   > **Note:**
   >
   > You can obtain *JAVA_HOME* reference from the `SetDomainEnv` script.

   a. Add RSA Security Provider to the top of the security file *JAVA_HOME*`/jre/lib/security/java.security`.

   b. update the sequence number for the remaining providers, as shown:

   ```
   security.provider.1=com.rsa.jsafe.provider.JsafeJCE
   security.provider.2=com.rsa.jsse.JsseProvider
   ```

2. Update WLS Pre-ClassPath Setting with FIPS specific jars. To do so:

   a. Set WLS `PRE_CLASSPATH` variable to point to `jcmFips.jar` and `sslj.jar`, which is in the *WL_HOME*`/server/lib/` directory.

   b. Export `PRE_CLASSPATH` by adding an entry in the `setDomainEnv.sh` script, which is in the *DOMAIN_HOME*`/bin/` directory. The following is a sample entry:

   ```
   PRE_CLASSPATH="WLS_HOME/server/lib/jcmFIPS.jar:WLS_HOME/server/lib/sslj.jar"
   export PRE_CLASSPATH
   ```

   Here, replace *WLS_HOME* with the absolute path of WLS_HOME in your environment after confirming that `jcmFIPS.jar` and `sslj.jar` exists in the location specified. This will set the PRE_CLASSPATH variable for the entire WLS Domain.

3. Restart the WebLogic Administrative Server and all Managed Servers.

A.3.5 Supported TLS Protocols and Cipher Suites by Oracle Unified Directory

TLS is a widely used protocol today by applications that entails data transmission over a network. The primary goal of the TLS protocol is to provide enhanced security and data integrity between two communicating applications.

Oracle Unified Directory supports protocols and cipher suites provided by Java Secure Socket Extension (JSSE). This section contains the following topics:

- Supported System Default TLS Protocols by Oracle Unified Directory
- Supported TLS Cipher Suites by Oracle Unified Directory
- Configuring JVM Cipher Suite

A.3.5.1 Supported System Default TLS Protocols by Oracle Unified Directory

Oracle Unified Directory supports TLS version 1.2 protocol by default.

TLS version 1.2 is the preferred protocol for TLS communication with Oracle Unified Directory.

A.3.5.2 Supported TLS Cipher Suites by Oracle Unified Directory

During a TLS handshake, the two communicating parties negotiate to determine which cipher suite they will use while transmitting messages back and forth.

Oracle Unified Directory implements the cipher suites (both the supported and the non-supported cipher suites) defined by the Oracle Software Security Assurance standards.

When you configure Oracle Unified Directory for secure communication, the server supports the use of TLS version 1.2 protocol and a list of ciphers with new authenticated encryption modes defined in TLS 1.2 with AES in Galois Counter Mode (GCM). Despite the enhancements, Oracle Unified Directory allows you to use older cipher suites to support users working on legacy platforms. However, it is recommended that you use stronger cipher suites.

Oracle Unified Directory supports a large number of cipher suites that are considered secure by the latest update of Java configuration. The tables in this section lists the cipher suites supported by Oracle Unified Directory in preference order.

> **Note:**
>
> You must note that the any cipher suite is only enabled if it is supported by the JVM you have deployed.

Table A-10 Default Enabled Cipher Suites

TLS Protocol	Cipher Suite
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_RSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_RSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
TLS version 1.2	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_DHE_DSS_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS version 1.2	TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
TLS version 1.2	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_DHE_DSS_WITH_AES_128_CBC_SHA

Table A-10 (Cont.) Default Enabled Cipher Suites

TLS Protocol	Cipher Suite
TLS version 1.2	TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_DH_DSS_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_DH_DSS_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_RSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_DH_DSS_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_RSA_WITH_AES_256_CBC_SHA256
TLS version 1.2	TLS_DH_DSS_WITH_AES_256_CBC_SHA256
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
TLS version 1.2	TLS_RSA_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_DH_DSS_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
TLS version 1.2	TLS_RSA_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_DH_DSS_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA
TLS version 1.2	TLS_DH_RSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256
TLS version 1.2	TLS_DH_RSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384
TLS version 1.2	TLS_DH_RSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
TLS version 1.2	TLS_DH_RSA_WITH_AES_256_CBC_SHA256
TLS version 1.2	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384

https://docs.oracle.com/en/java/javase/17/security/oracle-providers.html#GUID-FE2D2E28-C991-4EF9-9DBE-2A4982726313

A.3.5.3 Configuring JVM Cipher Suite

In some cases, you want to add to the Oracle Unified Directory system default protocols and ciphers, instead of completely overriding them. For such scenarios, Oracle Unified Directory provides a `jvm` keyword, which encapsulates all the Oracle Unified Directory system default cipher suites.

If you want to add a new cipher suite, for instance `SSL_DH_anon_WITH_DES_CBC_SHA` to the Oracle Unified Directory system default list, then you can specify the following cipher suites:

```
jvm, SSL_DH_anon_WITH_DES_CBC_SHA
```

The system will resolve the `jvm` keyword to system default cipher suites, and then add the new cipher suite `SSL_DH_anon_WITH_DES_CBC_SHA` at the end of the list.

Note that the `jvm` keyword can be used for both server side components (using `ssl-cipher-suite` property) and CLI tools (using `cipher_suite_sequence` property).

Sample properties files for CLI tools, which include the `jvm` keyword:

```
tls_protocols=TLSv1.1,TLSv1
cipher_suite_sequence=TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,\
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256,\
TLS_DHE_RSA_WITH_AES_128_CBC_SHA,\
TLS_DHE_RSA_WITH_AES_256_CBC_SHA,\
jvm,\
SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA,\
SSL_DH_anon_EXPORT_WITH_RC4_40_MD5,\
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA,\
TLS_EMPTY_RENEGOTIATION_INFO_SCSV,\
SSL_DH_anon_WITH_DES_CBC_SHA,\
SSL_DH_anon_WITH_RC4_128_MD5
```

A.3.6 Overview of Basic Encoding Rules

The Basic Encoding Rules (BER) are a set of Abstract Syntax Notation One encoding rules that define a specific way in which information may be encoded in a binary form.

BER is used as the underlying mechanism for encoding message.

This section contains the following topics:

- Understanding Basic Encoding Rules
- About BER Type
- About BER Length
- About BER Value
- Examples of Using BER Encoding

A.3.6.1 Understanding Basic Encoding Rules

Many network protocols are text-based, which has the advantages of being relatively easy to understand if you examine the network traffic, and you can often even interact with the target server by telnetting to it and typing in the appropriate commands.

However, there are disadvantages as well, including that they are generally more verbose and less efficient to parse than they need to be. On the other hand, other protocols use a binary encoding that is more compact and more efficient. LDAP falls into this category, and uses the ASN.1 (abstract syntax notation one) mechanism, and more specifically the BER (basic encoding rules) flavor of ASN.1. There are several other encoding rules (such as DER, PER, and CER) that fall under the ASN.1 umbrella, but LDAP uses BER.

This section discusses the subset of BER that is used by LDAP in particular and does not address other cases.

BER elements use a TLV structure, where TLV stands for "type", "length", and "value". That is, each BER element has one or more bytes (in LDAP, typically only a single byte) that indicates the data type for the element, one or more bytes that indicate the length of the value, and the encoded value itself (where the form of the encoded value depends on the data type), which can be zero or more bytes, as described in the following sections.

A.3.6.2 About BER Type

The BER type indicates the data type for the value of the element.

The BER specification provides several different data types, but the most commonly used by LDAP include `OCTET STRING` (which can be either a text string or just some binary data), `INTEGER`, `BOOLEAN`, `NULL`, `ENUMERATED` (like an integer, but where each value has a special meaning), `SEQUENCE` (an ordered collection of other elements, similar to an array), and `SET` (the same as a sequence, except that the order does not matter). There is also a `CHOICE` element, but it typically allows one of a few different kinds of elements.

The BER type is typically only a single byte, and this byte has data encoded in it. The two most significant bits (the two leftmost bits, because BER uses big endian/network ordering) are used to indicate the class for the element, using these possible class values:

- 00

  The universal class. Most BER elements have a universal type, so any element with a universal type specifies what kind of data it holds. Examples of universal types include 0x01 (`BOOLEAN`), 0x02 (`INTEGER`), 0x04 (`OCTET STRING`), 0x05 (`NULL`), 0x0A (`ENUMERATED`), 0x30 (`SEQUENCE`), and 0x31 (`SET`). The binary encodings for all of those type values have the leftmost two bits set to zero.

- 01

  The application-specific class. This class allows an application to define its own types that are consistent throughout that application. In this context, LDAP is considered an application. For example, when 0x42 appears in LDAP, it indicates an unbind request protocol op, because RFC 2251 section 4.3 (`https://tools.ietf.org/html/rfc2251#section-4.3`) states that the unbind request protocol op has a type of `[APPLICATION 2]`.

- 10

  The context-specific class. This class indicates that the type is specific to a particular usage within a given application. You can reuse the same type in different contexts within the same application if there is enough other information to determine which context is applicable in a given situation. For example, in the context of the credentials in a bind request protocol op, the context-specific type 0x80 is used to hold the bind password, but in the context of an extended operation it would be used to hold the request OID.

- 11

  The private class, not typically used in LDAP.

The next bit (the third from the left) is the primitive/constructed bit. If it is set to zero (off), then the element is considered primitive, and the value is encoded in accordance with the rules of that data type. If it is set to one (on), then it means that the value is constructed from zero or more other ASN.1 elements that are concatenated together in their encoded forms. For example, for the universal `SEQUENCE` type of 0x30, the binary encoding is `00110000` and the primitive/constructed bit is set to one indicating that the value of the sequence is constructed from zero or more encoded elements.

The final five bits of the BER type byte specify the value of that type, and they are treated as a simple integer value (where `00000` is zero, `00001` is one, `00010` is two, `00011` is three, and so on). The only special value is `11111`, which means that the type value is larger than can fit in the five bits allowed, and so multiple bytes are required. This value is not used in LDAP.

A.3.6.3 About BER Length

The second component in the TLV structure of a BER element is the length. This specifies the size in bytes of the encoded value.

For the most part, this uses a straightforward binary encoding of the integer value (for example, if the encoded value is five bytes long, then it is encoded as 00000101 binary, or 0x05 hex), but if the value is longer than 127 bytes then it is necessary to use multiple bytes to encode the length. In that case, the first byte has the leftmost bit set to one and the remaining seven bits are used to specify the number of bytes required to encode the full length. For example, if there are 500 bytes in the length (hex 0x01F4), then the encoded length will actually consist of three bytes: `82 01 F4`.

Be aware that there is an alternate form for encoding the length called the indefinite form. In this mechanism, only a part of the length is given at a time, similar to the chunked encoding that is available in HTTP 1.1. However, this form is not used in LDAP, as specified in RFC 2251 section 5.1 (https://tools.ietf.org/html/rfc2251#section-5.1).

A.3.6.4 About BER Value

The BER element contains the actual data of the element. Because BER is a binary encoding, the encodings can take advantage of that to represent the data in a compact form.

As such, each data type has its own encoded form:

NULL
The `NULL` element never has a value, and therefore the length is always zero.

OCTET STRING
The value of this element is encoded as a concatenation of the raw bytes of the data being represented. For example, to represent the string `Hello`, the encoded value would be `48 65 6C 6C 6F`. The value can have a length of zero bytes.

BOOLEAN
The value of this element is always a single byte. If all the bits in that byte are set to zero (0x00), then the value is `FALSE`. If one or more of the bytes is set to one, then the value is `TRUE`. As a result, there are 255 different ways to encode a `BOOLEAN` value of `TRUE`, but in practice it is generally encoded as 0xFF (that is, all the bits are set to one).

INTEGER
The value of this element is encoded as a binary integer in two's complement form. Although BER itself does not place a limit on the magnitude of the values that can be encoded, many software implementations have a cap of four or eight bytes (that is, 32-bit or 64-bit integer values), and LDAP generally uses a maximum of 4 bytes (which allows encoding values within the plus or minus 2 billion range). There is always at least one byte in the value.

ENUMERATED
The value of this element is encoded in exactly the same way as the value of an `INTEGER` element.

SEQUENCE
The value of this element is simply a concatenation of the encoded BER elements contained in the sequence. For example, to encode a sequence with two octet string elements encoding the text `Hello` and `there`, the encoded sequence value is `04 05 48 65 6C 6C 6F 04 05 74 68 65 72 65`. A sequence value can be zero bytes if there are no elements in the sequence.

SET
The value of this element is encoded in exactly the same way as the value of a `SEQUENCE` element.

A.3.6.5 Examples of Using BER Encoding

Review this example for encoding a `SEQUENCE` value had two complete BER elements concatenated together: the `OCTET STRING` representations of the strings `Hello` and `there`.

```
04 05 48 65 6C 6C 6F
04 05 74 68 65 72 65
```

In both of these cases, the first byte is the type (0x04, which is the universal primitive `OCTET STRING` type), and the second is the length (0x05, indicating that there are five bytes in the value). The remaining five bytes are the encoded representations of the strings `Hello` and `there`.

The following example encodes the integer value `3` using a context-specific type value of `5` instead of the universal `INTEGER` type:

```
85 01 03
```

The next example encodes an LDAP bind request protocol op as defined in RFC 2251 section 4.2 (https://tools.ietf.org/html/rfc2251#section-4.2). A simplified BNF representation of this element is as follows:

```
BindRequest ::= [APPLICATION 0] SEQUENCE {
     version                    INTEGER (1 .. 127),
     name                       OCTET STRING,
     authentication             CHOICE {
          simple                [0] OCTET STRING,
          sasl                  [3] SEQUENCE {
               mechanism        OCTET STRING,
               credentials      OCTET STRING OPTIONAL } } }
```

This example encodes a bind request using simple authentication for the user `cn=test` with a password of `password`. The complete encoding for this bind request protocol op is:

```
60 16 02 01 03 04 07 63 6E 3D 74 65 73 74 80 08 70 61 73 73 77 6F 72 64
```

In analysis, that string of bytes contains the following information:

- The first byte is 0x60 and it is the BER type for the bind request protocol op. It comes from the `[APPLICATION 0] SEQUENCE` portion of the definition. Because it is application-specific, then the class bytes are `01`, and because it is a `SEQUENCE`, it is constructed. Put that together with a type value of zero, the binary representation is `01100000`, which is 0x60 hex.
- The second byte is 0x16, which indicates the length of the bind request sequence. 0x16 hex is 22 decimal, and the number of bytes after the 0x16 is 22.
- The next three bytes are `02 01 03`, which is a universal `INTEGER` value of `3`. It corresponds to the version component of the bind request sequence, and it indicates that this is an LDAPv3 bind request.
- The next nine bytes are `04 07 63 6E 3D 74 65 73 74`, which is a universal `OCTET STRING` containing the text `cn=test`. It corresponds to the "name" component of the bind request sequence.

- The last component is `80 08 70 61 73 73 77 6F 72 64`, which is an element with a type of `context-specific primitive 0` and a length of eight bytes. As specified in the definition of the bind request protocol op, `context-specific` maps to the simple authentication type and that it should be treated as an `OCTET STRING`, and those eight bytes in the value do represent the encoded string `password`.

A.3.7 Authenticating Using CRAM-MD5 SASL Mechanism

The CRAM-MD5 Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server with a username and password in a manner that does not expose the clear-text password, so it is significantly safer than simple authentication or the PLAIN SASL mechanism when the connection between the client and the server is not secure.

The draft-ietf-sasl-crammd5-10 (`http://tools.ietf.org/html/draft-ietf-sasl-crammd5-10`) Internet Draft describes the CRAM-MD5 SASL mechanism. The process is as follows:

1. The client sends an message to the server with a bind request protocol op type using an authentication type of `SASL` with a mechanism name of `CRAM-MD5` and no credentials.
2. The server sends a bind response message back to the client with a result code of 14 (SASL bind in progress) and a server SASL credentials element including randomly-generated data (the challenge).
3. The client responds with a second SASL bind request message to the server with a mechanism name of `CRAM-M5`, and this time provides SASL credentials containing the authentication ID used to identify the user and an MD5 digest that is computed by combining the server-provided challenge with the clear-text password.
4. The server uses the authentication ID to identify the user, and then retrieves the clear-text password for that user (if the clear-text password cannot be obtained, then authentication will fail) and uses it to determine whether the provided digest is valid. The server will then send an appropriate response to the client (usually with a result of either `success` or `invalid credentials`) indicating whether the authentication was successful.

The CRAM-MD5 SASL mechanism is very similar to DIGEST-MD5 SASL mechanism, but it is somewhat weaker because CRAM-MD5 only includes random data from the server whereas DIGEST-MD5 includes random data from both the client and the server. DIGEST-MD5 also provides a provision for ensuring connection integrity, confidentiality, or both that CRAM-MD5 does not offer.

A.4 Glossary of Terms for Oracle Unified Directory

This glossary defines the terms that are used to describe LDAP and directory services, including terms that are specific to Oracle Unified Directory.

A.4.1 A

A.4.1.1 abandon operation

The LDAP abandon operation can be used to request that the server stop processing on an outstanding request. The abandon request protocol op is as follows:

```
AbandonRequest ::= [APPLICATION 16] MessageID
```

The message ID provided in the request is the message ID of the operation to abandon.

The abandon operation does not have a response, so there is no way for clients to know whether the abandon operation was successful. Similarly, if an operation was abandoned, then no response will be provided for it, so the client may wait indefinitely for a response that will never be sent. Both of these issues are addressed by the cancel extended operation.

Bind, unbind, abandon, and StartTLS extended operations cannot be abandoned.

A.4.1.2 abstract object class

An abstract object class is one that cannot be used directly in an entry but must be subclassed by either a structural object class or auxiliary object class. The subclasses will inherit any required attribute type, optional attribute type, or both attribute types as defined by the abstract class.

One of the most notable abstract object classes defined in LDAP is the `top` object class, which is the root class for virtually all other object classes defined in the server schema.

A.4.1.3 Abstract Syntax Notation One

Abstract Syntax Notation One (ASN.1) is a mechanism for encoding data in a binary form. It uses a TLV structure, in which each element has a type, length, and value. The type component is a data type that indicates what kind of information is stored in the element and indicates how the value should be encoded. The length component specifies the number of bytes in the value, and the value is the actual data held by the element.

Examples of ASN.1 elements include the following:

- Null

  Null elements do not hold any value. They are generally used as placeholders when an element is required but no value is needed.

- Octet string

  Octet string elements hold a set of zero or more octets (bytes) of data. It can be used for holding string or binary data.

- Boolean

  Boolean elements hold values that represent either `true` or `false`.

- Integer

  Integer elements hold values that represent integer values.

- Enumerated

  Enumerated elements hold values that represent integer values where each value has a specific meaning.

- Sequence

  Sequence elements are containers that hold zero or more other ASN.1 elements in a manner where the order of the elements is significant.

- Set

  Set elements are containers that hold zero or more other ASN.1 elements in a manner where the order of the elements is not significant.

> **Note:**
>
> ASN.1 is a general framework for binary encoding, but does not actually define how the data should be encoded. That is handled by an encoding rule, and there are several different kinds of ASN.1 encoding rules. LDAP uses the Basic Encoding Rules encoding, but other types include Distinguished Encoding Rules (DER), Canonical Encoding Rules (CER), and Packed Encoding Rules (PER).

A.4.1.4 access control

Access control provides a mechanism for restricting who can get access to various kinds of information in the Directory Server. You can use the access control provider to control several things, including:

- Whether a client can retrieve an entry from the server.
- Which attributes within the entry the client is allowed to retrieve.
- Which values of an attribute the client is allowed to retrieve.
- The ways in which the client can manipulate data in the directory.

A number of things can be taken into account when making access control decisions, including:

- The DN as whom the user is authenticated.
- The method by which the client authenticated to the server.
- Any groups in which that user is a member.
- The contents of the authenticated user's entry.
- The contents of the target entry.
- The address of the client system.
- Whether the communication between the client and server is secure.
- The time of day, the day of week, or both the time of day and day of week of the attempt.

See Controlling Access To Data for details on the access control syntax.

In addition to the access control subsystem, the directory server also provides a privilege that can be used to control what a user will be allowed to do. One of the privileges available is the `bypass-acl` privilege, which can be used to allow that client to bypass any restrictions that the access control subsystem would otherwise enforce.

A.4.1.5 access control instruction (ACI)

See access control rule

A.4.1.6 access control rule

An access control rule (also called an access control instruction, or ACI), is a rule which may be used to grant or deny a user or set of users access to perform some kind of operation in the server. The Directory Server access control policy comprises the complete set of access control rules defined in the server.

See Controlling Access To Data for more information about the syntax used for access control rules and the operations that can be allowed or denied using them.

A.4.1.7 access log

The Directory Server access log provides a mechanism for keeping track of every operation processed by the server, including every request received and response returned. It may also be used to obtain information about the internal operations performed within the server.

The directory server provides an extensible framework for implementing access loggers (as well as error log and debug log loggers). The default access control log implementation writes information to a log file with two records per operation. The first record reflects the request received from the client and the second provides information about the result of the operation processing.

All messages will include a common set of elements including:

- The time that the message was logged.
- The type of operation being processed.
- The connection ID of the client connection that requested the operation.
- The operation ID of the operation on that client connection.
- The message ID of the message used to request the operation.

For abandon operation, request log messages include the message ID of the operation to abandon. There is no response to an abandon operation, but the server will nevertheless log a result message indicating whether the abandon was successful and the processing time in milliseconds.

For add operation, request log messages include the distinguished name of the entry to add. The response log message may include the result code, diagnostic message, matched DN, the authorization ID for the operation, and the processing time in milliseconds.

For bind operation, request log messages include the authentication type (either `SIMPLE` or `SASL` followed by the mechanism name) and the bind DN. The response log message may include the result code, diagnostic message, matched DN, authentication ID, authorization ID, and processing time in milliseconds.

For compare operation, request log messages include the target entry DN and the attribute type. The response log message may include the result code, diagnostic message, matched DN, authorization ID, and the processing time in milliseconds.

For delete operation, request log messages include the target entry DN. The response log message may include the result code, diagnostic message, matched DN, authorization ID, and the processing time in milliseconds.

For extended operation, request log messages include the object identifier for the extended request. The response log message may include the OID of the extended response, the result code, diagnostic message, matched DN, and the processing time in milliseconds.

For modify operation, request log messages include the target entry DN. The response log message may include the result code, diagnostic message, matched DN, authorization ID, and the processing time in milliseconds.

For modify DN operation, request log messages include the target entry DN, the new RDN, a flag indicating whether to delete the old RDN values, and the new superior DN. The response log message may include the result code, diagnostic message, matched DN, authorization ID, and the processing time in milliseconds.

For search operation, request log messages include the search base DN, search scope, LDAP search filter, and search attributes. The response log message may include the result code, number of entries returned, diagnostic message, matched DN, authorization ID, and the processing time in milliseconds.

For unbind operation, the request message will simply indicate that an unbind request has been received. There is no response to an unbind request, and no result log message.

For connect operation, log messages include the server and client IP address and protocol. When the client connects over a secure channel, if the `log-connection-details` flag is set to `true`, additional details such as TLS version, negotiated cipher suite, is logged in a separate line. You can map other logs for the request using `conn=value`, where value is the connection ID.

A.4.1.8 account expiration

Account expiration is a component of the Directory Server password policy that may be used to indicate that an account is no longer able to be used beyond a given date. This feature may be useful for creating temporary user accounts (for example, for use by contractors, interns, or other temporary workers) that will expire after a specified date.

Account expiration may be enabled by adding the `ds-pwp-account-expiration-time` operational attribute to the target user's entry. The value for this attribute should be a time stamp in generalized time format that specifies the time that the account should expire. Once the account expiration time has passed, the user will no longer be allowed to authenticate to the server.

A.4.1.9 account lockout

Account lockout is a component of the Directory Server password policy that may be used to lock user accounts after too many failed bind attempts. Once an account has been locked, that user will not be allowed to authenticate. The lockout may be temporary (automatically ending after a specified period of time) or permanent (remaining in effect until an administrator resets the user's password).

A.4.1.10 account status notification

An account status notification is a mechanism that can be used to provide indication that a user account has changed in a manner that is significant regarding the server's password policy.

The types of account status notifications available for use in the server include:

- When the user's account has been account lockout
- When the user's account has been account lockout
- When the user's account has been unlocked by an administrator
- When the user's account has been manually disabled or reenabled by an administrator
- When the user's account expiration
- When the user's password expiration or is about to expire
- When the user's password has been password reset
- When the user's password has been changed by the end user

The directory server provides an extensible framework for handling account status notifications. The default handler writes messages to the server's error log, but the framework can be used to send email messages or take other actions that may be desired.

A.4.1.11 account usability control

The account usability control provides a pair of request and response controls that can be used to determine whether a user account may be used for authenticating to the server.

The request control has an OID of 1.3.6.1.4.1.42.2.27.9.5.8 and does not include a value. It should only be included in search operation messages.

The corresponding response control has an OID of 1.3.6.1.4.1.42.2.27.9.5.8 (the same as the request control), and it will be included in any search result entry messages for a search request that includes the account usability request control.

The value for the account usability response control is encoded as follows:

```
ACCOUNT_USABLE_RESPONSE ::= CHOICE {
     is_available           [0] INTEGER, -- Seconds before expiration --
     is_not_available       [1] MORE_INFO }

     MORE_INFO ::= SEQUENCE {
     inactive               [0] BOOLEAN DEFAULT FALSE,
     reset                  [1] BOOLEAN DEFAULT FALSE,
     expired                [2] BOOLEAN DEFAULT_FALSE,
     remaining_grace        [3] INTEGER OPTIONAL,
     seconds_before_unlock  [4] INTEGER OPTIONAL }
```

If the user account is available, then the control will include the number of seconds until the user's password expires, or -1 if password expiration is not enabled. If the user's account is not available, then the control will provide the reason it is unavailable.

For an example of using this control in a search request, see Searching Using the Account Usability Request Control.

A.4.1.12 ACID

ACID is an acronym that stands for Atomicity, Consistency, Isolation, and Durability. This term is standard database terminology that refers to the characteristics that can be achieved using the transaction nature of the database. These elements include:

- Atomi city

  Each transaction performed in the database is atomic. That is, it either completely succeeds or completely fails. It never partially succeeds such that some changes that are part of the transaction are applied while others are not.

- Consistency

  The database is always in a consistent state such that the integrity of its contents will be preserved. It should not be possible for a successful or failed transaction to leave the database in an inconsistent state.

- Isolation

  The operations performed as part of a transaction will be isolated from other operations performed in the database at the same time. If one transaction is used to make several changes to database contents, then it should not be possible for another transactional operation to see the effects of those changes until they have been committed.

- Durability

Any transaction that the database has reported as complete and committed successfully is guaranteed to be on persistent storage. Even if the directory server, or the underlying JVM, operating system, or hardware should fail the instant after the notification of the successful commit, then that change will not be lost.

The Berkeley DB Java Edition used as the data store for the primary back end provides full support for ACID compliance, although it also provides methods for relaxing its compliance to these constraints if desirable for performance reasons. The directory server exposes some of this flexibility, particularly regarding configuring how durable the changes will be (for example, it is possible to configure the server so that changes are not immediately flushed to disk, which may allow better write performance but could cause the loss of one or more changes if you have a hardware or software failure).

A.4.1.13 add operation

The LDAP add operation can be used to create an entry in the Directory Server. The add request protocol op is defined as follows:

```
AddRequest ::= [APPLICATION 8] SEQUENCE {
     entry           LDAPDN,
     attributes      AttributeList }
```

The elements included in this request include the distinguished name of the entry to add and the set of attributes to include in that entry.

The response to an LDAP add operation is an LDAP result element, defined as follows:

```
AddResponse::= [APPLICATION 9] LDAPResult
```

A.4.1.14 alias

An alias is a special type of entry that references another entry in the server, much like a symbolic link in a UNIX file system. It should include the `alias` object class and the `aliasedObjectName` attribute with a value equal to the DN of the entry that it references.

Aliases are primarily used for search operation. In particular, the search request includes an element that specifies the dereference policy that should be used when aliases are encountered. The allowed dereference policy values include:

- neverDerefAliases

  The server should never dereference alias entries.

- derefInSearching

  The server should dereference any alias entries that it finds in the possible set of search result entries, but if the search base DN specifies an alias entry it will not be de referenced.

- derefFindingBaseObj

  The server should dereference the search base entry if it is an alias, but it will not dereference any aliases within the possible set of search result entries.

- derefAlways

  The server should dereference any aliases encountered, whether in the search base entry or in the possible set of search result entries.

> **Note:**
>
> Aliases are an optional part of the LDAPv3 protocol, and the directory server does not currently support them.

A.4.1.15 AND search filter

An AND search filter is a type of LDAP search filter that is intended to serve as a container that holds zero or more other search filters. In order for an entry to match an AND filter, it must match all of the filters contained in that AND filter.

AND filters may be represented as a string by enclosing the entire filter in parentheses and placing an ampersand just after the opening parenthesis. For example, a filter of `(&(objectClass=person)(uid=john.doe))` represents an AND search filter that embeds the `(objectClass=person)` and `(uid=john.doe)` equality filters.

An AND filter that does not contain any embedded filters is called an LDAP true filter. The string representation for an LDAP true filter is an ampersand `(&)`, and LDAP true filters will always match any target entry.

A.4.1.16 anonymous bind

An anonymous bind is a type of bind operation using simple authentication with a zero-length bind DN and a zero-length password. It may be used to destroy any previous authentication performed on a connection and return it to an unauthenticated state.

Be aware that there is an ANONYMOUS SASL mechanism that has the same effect, but in general the term "anonymous bind" refers to the simple bind operation with no DN and password.

A.4.1.17 ANONYMOUS SASL mechanism

The ANONYMOUS SASL mechanism is a type of Simple Authentication and Security Layer authentication mechanism. It is different from other SASL mechanisms in that it is used to create an unauthenticated session, and will destroy any previous authentication that may have been performed on the connection.

The ANONYMOUS SASL mechanism provides the ability to include trace information in the request that may be included in the server's access log. This trace information can provide information about the client performing the bind, although because no authentication is performed the validity of the trace information cannot be guaranteed.

A.4.1.18 approximate index

An approximate index is a type of index that is used to efficiently identify which entries are approximately equal to a given assertion value. An approximate index can be maintained only for attributes that have a corresponding approximate matching rule. That matching rule are used to normalized value to use as index keys, and the value for that key is the ID list containing the entry ID of the entries with values that are approximately equal to that normalized value.

A.4.1.19 approximate search filter

An approximate search filter is a type of LDAP search filter that can be used to identify entries that contain a value for a given attribute that is approximately equal to a given assertion value. The server will use an approximate matching rule to make the determination.

The string representation of an LDAP approximate filter comprises an opening parenthesis followed by the attribute name, a tilde, an equal sign, the attribute value, and the closing parenthesis. For example, an equality filter of `(givenName~=John` will match any entry in which the `givenName` attribute contains a value that is approximately equal to `John`.

A.4.1.20 ASN.1

See Abstract Syntax Notation One.

A.4.1.21 assertion value

An assertion value is the value of an attribute value assertion. The assertion value is provided to a matching rule to make a determination about the attribute value of a specified attribute.

A.4.1.22 attribute

An attribute is a named set of values. An attribute has an attribute description, which contains the name of that attribute (which links it to an attribute type) and an optional set of attribute option, and a collection of one or more values.

An entry contains a collection of attributes. It is possible for an entry to have multiple attributes with the same attribute type but different sets of options.

A.4.1.23 attribute description

An attribute description is used to identify a given attribute in an entry. An attribute description contains a name or OID that ties it to an attribute type and zero or more attribute option. If the attribute description contains any attribute options, then they are separated from the attribute name/OID by a semicolon, and a semicolon is also used to separate individual attribute options if there is more than one option in the attribute description.

A.4.1.24 attribute option

An attribute option is a kind of tag that provides additional information about the way that an attribute should be interpreted. An attribute description consists of the attribute name or object identifier followed by zero or more attribute options. If there are attribute options, then they are separated from the attribute name and from each other using semicolons. For example, in the attribute description `userCertificate;binary`, the attribute name is `userCertificate` and the attribute option is `binary`.

Attribute options can be used for several purposes, including providing information about how the server should treat that attribute (for example, the binary encoding option as described in RFC 4522 (`http://www.ietf.org/rfc/rfc4522.txt`)) They may also be provided for the benefit of clients in some form (for example, the language tag options as described in RFC 3866 (`http://www.ietf.org/rfc/rfc3866.txt`), which make it possible to provide an attribute value in different languages).

A.4.1.25 attribute syntax

An attribute syntax is a schema element that defines a kind of data type that is used to dictate the kind of information that may be stored in an attribute value. Any attempt to store an attribute value that violates the syntax for the associated attribute type should be rejected.

Common attribute syntaxes include:

- Binary

  Can hold any kind of data, whether textual or not, that should be compared on a byte-for-byte basis.

  > **Note:**
  >
  > The binary syntax has been deprecated in favor of the octet string syntax.

- Boolean

  Can hold values of either `TRUE` or `FALSE`.

- Directory String

  Can hold any kind of string value (technically, binary values are allowed as well, but directory string values are typically strings).

- Distinguished Name

  Can hold values that are valid distinguished name.

- Generalized Time

  Can hold values that contain time stamps of varying precision (anywhere from an hour to a fraction of a second) including time zone information. For example, the value `20070525222745Z` represents a time stamp of May 25, 2007 at 10:27:45 PM in the UTC time zone.

- IA5 String

  Can hold values that contain ASCII strings (that is, use of non-ASCII characters is not allowed).

- Integer

  Can hold integer values. Positive, negative, and zero values are allowed.

- Octet String

  Can hold any kind of data that should be compared on a byte-for-byte basis.

- Postal Address

  Can hold a multi-line address, in which the lines of the address should be separated by dollar signs.

- Printable String

  Can hold a string containing any combination of printable characters. Printable characters include all uppercase and lowercase ASCII letters, the numeric digits, the space character, and the symbols `'()+,-.=/:?`.

- Telephone Number

Can hold telephone number values.

The set of attribute syntaxes defined in the server may be determined by retrieving the `ldapSyntaxes` attribute of the subschema subentry. For more information about attribute syntaxes, see Overview of Attribute Syntaxes.

A.4.1.26 attribute type

An attribute type is a schema element that correlates an object identifier and a set of names with an attribute syntax and a set of matching rule.

The components of an attribute type definition include:

- An OID used to uniquely identify the attribute type.
- A set of zero or more names that can be used to more easily reference the attribute type.
- An optional equality matching rule that specifies how equality matching should be performed on values of that attribute. If no equality matching rule is specified, then the default equality rule for the associated attribute syntax will be used. If the associated syntax does not have a default equality matching rule, then equality operations will not be allowed for that attribute.
- An optional ordering matching rule that specifies how ordering operations should be performed on values of that attribute. If no ordering matching rule is specified, then the default ordering rule for the associated attribute syntax will be used. If the associated syntax does not have a default ordering matching rule, then ordering operations will not be allowed for that attribute.
- An optional substring matching rule that specifies how substring matching should be performed on values of that attribute. If no substring matching rule is specified, then the default substring rule for the associated attribute syntax will be used. If the associated syntax does not have a default substring matching rule, then substring operations will not be allowed for that attribute.
- An optional syntax OID that specifies the syntax for values of the attribute. If no syntax is specified, then it will default to the directory string syntax.
- A flag that indicates whether the attribute is allowed to have multiple values.
- An optional attribute usage string indicating the context in which the attribute is to be used.
- An optional flag that indicates whether the attribute can be modified by external clients.

The set of attribute types defined in the server may be determined by retrieving the `attributeTypes` attribute of the subschema subentry. For more information about attribute types, see Understanding Attribute Types.

A.4.1.27 attribute usage

An attribute type attribute usage defines the contexts in which it may be used. There are four types of attribute usage:

- userApplications

  This should be used for all attribute types that are intended for use in holding user-defined data.

- directoryOperation

  This should be used for attribute types that are used for behind-the-scenes processing within the server.

- distributedOperation

  This should be used for attribute types that store operational data that need to be distributed (that is, replication) throughout the directory environment.

- dSAOperation

  This should be used for attribute types that store operational data that should be stored only in one server and should not be replicated throughout the directory environment.

Attributes with a usage of `userApplications` are known as user attribute. Attributes with a usage of `directoryOperation`, `distributedOperation`, or `dSAOperation` are known as operational attribute.

A.4.1.28 attribute value

An attribute value describes an element of actual data held by an attribute. An attribute may have multiple values, if allowed by the associated attribute type. The way that the server should interact with the values of that attribute is governed by that attribute's attribute syntax and matching rule.

A.4.1.29 attribute value assertion

An attribute value assertion (AVA) is a combination of an attribute description and an attribute value. The assertion value is used with a matching rule to make the determination. If the matching rule is an equality matching rule, then it will be used to determine whether the attribute contains a given value. If it is an ordering matching rule, then the AVA will be used to determine whether the attribute contains a value that is greater than or equal to, or less than or equal to, the assertion value. If it is an approximate matching rule, then the AVA will be used to determine whether the attribute contains a value that is approximately equal to the assertion value. Substring matching is more complex and uses a substring assertion rather than a simple assertion value.

Attribute Value assertions are used in LDAP compare operation, as well as equality search filter, greater than or equal to search filter, less than or equal to search filter, and approximate search filter search filters.

A.4.1.30 audit log

The audit log is a special type of access log that is used to log information about all changes that are made in the server. It provides a log of those changes in LDAP Data Interchange Format form so that administrators can see exactly what changes were made. This information can be used for diagnostic purposes when investigating a problem, to help better understand the kinds of changes that an application might make in the directory, or to help collect information about changes for replay to an alternate repository.

The name "audit log" is a legacy term referring to its use in the Netscape Directory Server. Do not confuse audit log with a log that could be used for security auditing, because it only records changes to directory data and does not keep track of things like successful or failed authentication attempts. However, you can often use the combination of the content from the traditional access log and the audit log to obtain this kind of information. If desired, an administrator could also provide a custom access logging implementation to keep track of any kind of desired information.

A.4.1.31 authentication

Authentication is the process whereby a client identifies itself to the directory server and provides proof of its identity. In LDAP, this is performed with a bind operation.

The authentication process has two phases:

- Identification

  The client identifies itself to the server in some way. In simple authentication, the DN provided in the bind request is used for this purpose. In Simple Authentication and Security Layer authentication, the identity of the client is obtained through some other means (for example, using a certificate, a Kerberos principal, or some other kind of identifier).

- Verification of Identity

  The client must provide sufficient proof that it is who it has identified itself to be. In simple authentication, this is done through the password. In SASL authentication, this verification is obtained in a manner specific to the associated mechanism (it may be a password, or it may be a certificate or some other form of proof).

Some authentication mechanisms may be considered stronger than others. For example, simple authentication may be considered less trustworthy if the client has a password that is easy to guess or obtain through some other means, whereas authentication using a certificate or Kerberos credentials might be considered much stronger and harder to forge. The directory server's access control implementation may be configured to take the client's authentication mechanism into account when determining whether a requested operation will be allowed.

A.4.1.32 authentication ID

An authentication ID is an identifier that is used by a client to identify itself to the Directory Server for certain kinds of Simple Authentication and Security Layer mechanisms (for example, CRAM-MD5 SASL mechanism, DIGEST-MD5 SASL mechanism, and PLAIN SASL mechanism). It can be used to allow a client to identify itself with a username (or other friendly identifier) rather than a distinguished name.

In most cases, an authentication ID should be specified in one of the following forms:

- The string `dn:` followed by the distinguished name of the target user (or just the string `dn:` if the authentication identity should be that of the anonymous user).
- The string `u:` followed by a username used to identify the user. An identity mapper will be used to map the provided username to the corresponding user entry.

A.4.1.33 authentication password syntax

The authentication password syntax defines a standard method for encoding a user password for storage in the server, ideally in a manner that makes it difficult or impossible to determine the clear-text value of that password.

RFC 3112 (`http://www.ietf.org/rfc/rfc3112.txt`) describes the authentication password syntax, which defines the `authPassword` attribute type and a corresponding `authPasswordObject` auxiliary object class that allows the use of that attribute.

The basic form of a password encoded using the authentication password syntax is:

scheme `$`*authInfo* `$` *authValue*

where *scheme* is the name of the scheme used to encode the value, *authInfo* is some kind of modifier (for example, a salt) used in the encoding process, and *authValue* is the encoded password information. For example, the value `SHA1$RzqH67DY3uQ=$atAcDs1eS+IJwPy7V4UDXEoBrDI=` is encoded using the authentication password syntax The scheme is `SHA1`, the authInfo element is `RzqH67DY3uQ=`, and the authValue element is `atAcDs1eS+IJwPy7V4UDXEoBrDI=`.

The authentication password schemes supported by the directory server include the following:

- MD5

  Uses the MD5 message digest.

- SHA1

  Uses the SHA-1 variant of the Secure Hash Algorithm.

- SHA256

  Uses the 256-bit SHA-2 variant of the Secure Hash Algorithm.

- SHA384

  Uses the 384-bit SHA-2 variant of the Secure Hash Algorithm.

- SHA512

  Uses the 512-bit SHA-2 variant of the Secure Hash Algorithm.

A.4.1.34 authorization

Authorization is the process of determining whether a user will be allowed to perform a requested operation. A number of server components may be involved in the authorization process, including:

- The access control handler.
- The privilege subsystem.
- The password policy.
- Custom plug-in installed in the server.

A.4.1.35 authorization ID

An authorization ID is an identifier that is used by a client to indicate that one or more operations should be performed under the authority of an alternate identity. This alternate authorization identity can last for a single operation (when used with the proxied authorization control) or for the entire duration of an authentication session (when used with an appropriate SASL mechanism, like DIGEST-MD5 SASL mechanism, GSSAPI SASL mechanism, or PLAIN SASL mechanism).

In most cases, an authorization ID should be specified in one of the following forms:

- The string `dn:` followed by the distinguished name of the target user (or just the string `dn:` if the authorization identity should be that of the anonymous user).
- The string `u:` followed by a username used to identify the user. An identity mapper maps the provided username to the corresponding user entry.

The ability for a client to use an alternate authorization identity is controlled by the `proxied-auth` privilege. In some cases, additional access control rights may also be required.

A.4.1.36 authorization identity control

The authorization identity controls are a pair of request and response controls defined in RFC 3829 (`http://www.ietf.org/rfc/rfc3829.txt`) that can be used with a bind operation to allow the client to learn the authorization identity for the client connection.

The authorization identity request control has an object identifier of `2.16.840.1.113730.3.4.16` and does not have a value. The authorization identity response

control has an OID of `2.16.840.1.113730.3.4.15` and the value of that control should be a string representing the authorization identify for that connection (or an empty string if the authorization identity is that of the anonymous user). The response control should only be included in the response if the authentication was successful.

> **Note:**
>
> The authorization identity controls are only allowed for use with the LDAP bind operation, and you cannot use them after the client has authenticated. You can use the "Who Am I?" extended operation to obtain the authorization identity at any time after the bind has completed.

For an example of using this control in a search request, see Searching Using the Authorization Identity Request Control.

A.4.1.37 auxiliary object class

An auxiliary object class is one that does not define the core type of an entry, but defines additional characteristics of that entry. An entry can contain zero or more auxiliary object classes. The set of auxiliary classes allowed for use in an entry may be controlled by a DIT content rule associated with that entry's structural object class.

A.4.1.38 AVA

See attribute value assertion

A.4.2 B

A.4.2.1 back end

A Directory Server back end provides a repository for storing data and a set of logic for interacting with that data. A back end will typically contain some kind of database and may maintain a set of index that allows the back end to quickly locate entries for various operations. All back ends will have the following qualities:

- A back end ID, which uniquely identifies that back end among all other back ends in the server.
- A set of one or more base distinguished name that indicate the data that the back end holds.
- A writability mode, which indicates whether the back end will accept write operations.

The logic provided by the back end includes:

- A method for determining whether a given entry exists, based on its DN
- A method for retrieving an entry, based on its DN
- A method of adding a new entry to the database (as part of processing an LDAP add operation)
- A method for removing an existing entry from the database (as part of processing an LDAP delete operation)

- A method for replacing an entry in the database (as part of processing an LDAP modify operation)
- A method for renaming an entry in the database (as part of processing an LDAP modify DN operation)
- A method for processing an LDAP search operation
- A method for exporting the contents of the database in LDAP Data Interchange Format form
- A method for importing data in LDAP Data Interchange Format form into the database
- A method for performing a backup of the data
- A method for performing a restore of a previous backup

A.4.2.2 backup

A backup is a transportable representation of the data in a Directory Server back end. Each back end is responsible for controlling whether it is possible to back up its contents, and ensuring that the backup information is suitable to be restore at a later time.

> **Note:**
>
> The term *back up* is a verb (the action of backing up the contents of the back end) and *backup* is a noun (what you get when you perform a back up).

There are several reasons that a back end may not provide a backup mechanism. Some reasons include:

- The back end only contains temporary, point-in-time information that does not make sense to archive or attempt to restore at a later time (for example, the root DSE or the monitor back ends).
- The back end stores its information in a remote repository that is not directly available to be archived. In cases like this, the external repository will likely have its own backup and restore mechanism.

The primary back end used by the directory server is one that uses the Berkeley DB Java Edition as its underlying database and that back end provides complete backup and restore capabilities. The backup mechanism is also very portable and can be transported across different platforms and different filesystem locations, and it is suitable for use as a binary copy mechanism.

A.4.2.3 base64 encoding

Base64 encoding is a way of representing binary data in a text-only form. It is commonly used in LDAP Data Interchange Format for values containing non-ASCII characters, or for values that could otherwise be ambiguous (for example, values that begin or end with spaces). It is also frequently used to encode certificate contents or the output of message digests like MD5 or Secure Hash Algorithm. Section 5.2 of RFC 1341 (`http://www.ietf.org/rfc/rfc1341.txt`) describes base64 encoding.

The basic principle of base64 encoding is that it defines a 64-character alphabet containing the following characters in the given order:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

Each of those characters is assigned a numeric value between 0 and 63 based on its position in the list (that is, `A` is 0, `B` is 1, `C` is 2,... `+` is 62, and `/` is 63). A value is broken up into six-bit segments, and each of those six bits is converted into a numeric value between 0 and 63 and replaced with the specified character from the alphabet given above. This means that every three bytes of a binary value is converted into four characters from the base64 alphabet. If the length of the binary value is not a multiple of three bytes, then it is zero-padded and either one or two equal signs are appended to the base64-encoded value.

A.4.2.4 Basic Encoding Rules

The Basic Encoding Rules (BER) are a set of Abstract Syntax Notation One encoding rules that define a specific way in which information may be encoded in a binary form. It is used as the underlying mechanism for encoding message. See Overview of Basic Encoding Rules

A.4.2.5 BER

See Basic Encoding Rules

A.4.2.6 Berkeley DB Java Edition

The Berkeley DB Java Edition (also referred to as "Berkeley DB JE", "BDBJE", or "JE") is a pure Java database designed by Sleepycat Software, which was purchased by the Oracle Corporation. It provides a highly-scalable, high-performance, transactional B-Tree database, with support for full ACID semantics and it is used as the primary database for storing user data.

The directory server provides a back end that uses the Berkeley DB Java Edition for storing its information. This back end is often called the "JE Backend" or simply "JEB". It uses a Berkeley DB Java Edition environment that consists of multiple individual databases. The id2entry database provides a mechanism for mapping entry ID values to entry contents. Other databases serve as index that can be used to quickly find entry contents for processing various types of operations.

A.4.2.7 binary copy

Binary copy refers to the process of performing a backup of a Directory Server back end of one server instance and restore that back end into another instance of the server. This can provide a fast disaster recovery mechanism and can also be used as a replica initialization mechanism.

Not all Directory Server back ends necessarily support the use of binary copy, and those that do may not support it in all circumstances. The primary back end type used by the directory server is based on the use of the Berkeley DB Java Edition, and it does support the use of the binary copy mechanism, including across different operating systems and CPU architectures, and with different filesystem locations. However, it does require that both servers have the same set of base distinguished name and the same types of index defined.

A.4.2.8 bind operation

The LDAP bind operation can be used to authenticate to the Directory Server. There are two basic types of bind operations:

- A simple bind operation, which uses simple authentication involving a bind DN and password to authenticate to the server.

- A SASL bind operation, which uses the Simple Authentication and Security Layer to authenticate the client, which can use a variety of types of credentials based on the selected SASL mechanism.

The bind request protocol op is defined as follows:

```
BindRequest ::= [APPLICATION 0] SEQUENCE {
     version                 INTEGER (1 ..  127),
     name                    LDAPDN,
     authentication          AuthenticationChoice }

AuthenticationChoice ::= CHOICE {
     simple                  [0] OCTET STRING,
                                 -- 1 and 2 reserved
     sasl                    [3] SaslCredentials,
     ...  }

SaslCredentials ::= SEQUENCE {
     mechanism               LDAPString,
     credentials             OCTET STRING OPTIONAL }
```

The elements of the request include:

- The LDAP protocol version. Allowed values are `2` and `3`, although LDAPv2 has been classified as a historical protocol and is no longer recommended for use.
- The bind DN. This is always used for simple authentication (although it may be a zero-length string for anonymous simple authentication), and is generally not used for SASL authentication.
- The credentials. The type of credentials provided vary based on the authentication type.
  - For simple authentication, the credentials should be the password for the target bind DN, or an empty string for anonymous simple authentication.
  - For SASL authentication, the credentials should include the name of the SASL mechanism to use, and may optionally include encoded credential information appropriate for the SASL mechanism.

The response to an LDAP bind operation is defined as follows:

```
BindResponse ::= [APPLICATION 1] SEQUENCE {
     COMPONENTS OF LDAPResult,
     serverSaslCreds    [7] OCTET STRING OPTIONAL }
```

This indicates that the bind response will include the elements in the LDAP result object and may also include a set of server SASL credentials if appropriate for the authentication type.

A.4.3 C

A.4.3.1 cancel extended operation

The LDAP Cancel extended operation is an extended operation that provides a function similar to the core LDAP abandon operation in that it can be used to request that the server stop processing on an operation in progress. The primary advantages of the Cancel extended operation over the abandon operation are that both the cancel request and the operation being canceled are guaranteed to get a response, whereas there is no response for the abandon request and there may not be a response for the operation being abandoned.

The Cancel extended operation is defined in RFC 3909 (http://www.ietf.org/rfc/rfc3909.txt). The value of the Cancel Request extended operation is encoded as follows:

```
cancelRequestValue ::= SEQUENCE {
     cancelID        MessageID
                     -- MessageID is as defined in [RFC2251]
}
```

A.4.3.2 CDDL

See Common Development and Distribution License.

A.4.3.3 certificate

A certificate is an element of public key cryptography that may be used to perform asymmetric encryption. In particular, a certificate consists of a pair of keys (called the "public key" and the "private key", respectively) that are linked so that any data encrypted using the public key can be decrypted using the private key. With many public key algorithms, like RSA, the reverse is also true so that any data encrypted with the private key can be decrypted using the public key.

The term certificate has different meanings, based on the context in which it is used. Often, it refers to only the public key (in particular, whenever the server presents its certificate to the client, or if a client presents its certificate to the server, then only the public key is included). However, in other cases, it does include the private key (i.e., the server will require the use of the private key to establish a secure communication channel with the client, and the client will need access to its private key to send its own certificate to the server).

Certificates have two primary uses in the directory server. The first is for providing a secure communication mechanism, generally through the Secure Sockets Layer or StartTLS extended operation. In this case, the negotiation process involves the client encrypting information using the server's public key so that only the server can decrypt it using its public key and that information will not be exposed to any third party that might be able to observe the communication. Certificates may also be used for data signing, in which case the server will encrypt information using its private key, and clients will know that the data is legitimately from the server if it can be decrypted using the server's public key.

A.4.3.4 certificate mapper

A certificate mapper provides the logic required to identify a user in the Directory Server that corresponds to a provided client certificate. The mapping may use any of the information contained in the certificate, although many certificate mappers are based primarily on the certificate's subject (the name of the certificate, which comprises several attribute-value pairs and looks very much like an LDAP distinguished name.

A.4.3.5 chaining

Chaining provides a mechanism for making data in a remote Directory Server instance appear as if it is part of the local server. That is, chaining is used to present a part of the directory information tree (DIT) using data from another server. Any request that the server receives for data in a chained portion of the DIT will be transparently forwarded to the server that actually contains the request.

A.4.3.6 changelog

A changelog is a special kind of database that is used to keep track of the changes that occur in a Directory Server instance. There are two different kinds of changelogs:

- A replication changelog stores change information in a format needed for replication.
- An LDAP-accessible changelog that represents its data in the format specified in draft-good-ldap-changelog that allows clients to learn about the changes that have occurred in the directory environment.

A.4.3.7 cn=Directory Manager

See directory manager.

A.4.3.8 collective attribute

A collective attribute is a special type of virtual attribute that is defined in RFC 3671 (http://www.ietf.org/rfc/rfc3671.txt). Collective attributes enable you to define values that are assigned to attributes based on an entry's membership in a subentry.

A.4.3.9 Common Development and Distribution License

The Common Development and Distribution License (CDDL) is an OSI-approved (http://www.opensource.org/) open source license which is used by the OpenDS project, on which Oracle Unified Directory.

The CDDL is a file-based license, which means that any changes to files contained in the project need to remain licensed under the CDDL. New files, however, may be licensed under any license chosen by the author (including closed-source licenses). The CDDL is based on the Mozilla Public License (MPL) and includes a patent grant clause so that any technology covered by patents will be granted to other projects using the code.

The CDDL license contents may be found at http://www.opensource.org/licenses/cddl1.php.

A.4.3.10 compare operation

The LDAP compare operation can be used to determine whether a specified entry contains a given attribute value. The compare request protocol op is defined as follows:

```
CompareRequest ::= [APPLICATION 14] SEQUENCE {
     entry           LDAPDN,
     ava             AttributeValueAssertion }

AttributeValueAssertion ::= SEQUENCE {
     attributeDesc   AttributeDescription,
     assertionValue  AssertionValue }
```

The elements of the request include the following:

- The DN of the entry in which the comparison is to be made.
- The name of the attribute in which the comparison is to be made.
- The assertion value to try to find in the specified attribute.

The response to an LDAP compare operation is an LDAP result element as defined below:

```
CompareResponse ::= [APPLICATION 15] LDAPResult
```

A.4.3.11 connection handler

A connection handler is a component of the Directory Server that is responsible for accepting connections from clients, reading and parsing requests submitted by the clients, ensuring that they are processed by the server, and sending the corresponding responses back to the client. The connection handler manages all communication with the client and therefore needs to implement support for the associated protocol.

The directory server currently provides connection handlers capable of communicating using Lightweight Directory Access Protocol and Java Management Extensions, as well as a special connection handler for internal use that may be used to allow components of the server (like plug-in and other kinds of extensions) to perform operations. The server also provides an extensible connection handler API that may be used to implement support for additional network protocols.

A.4.3.12 connection ID

A connection ID is a unique integer identifier that is assigned to each connection maintained within the Directory Server. It is used primarily for logging purposes, so that it is possible to correlate the various operations performed on a given connection.

The connection ID counter starts at zero for the first connection received by the server and increments by one for each additional connection. The counter is reset whenever the server is restarted.

Internal connections, which are used for processing internal operations, are assigned negative values to distinguish them from connections from external clients.

A.4.3.13 control

An LDAP control is an element that may be included in an message. If it is included in a request message, it can be used to provide additional information about the way that the operation should be processed. If it is included in the response message, it can be used to provide additional information about the way the operation was processed.

Examples of LDAP controls include:

- account usability control - This is a pair of request and response controls that indicate whether an account can authenticate to the server.
- authorization identity control - This is a pair of request and response controls that may be used to determine the authorization identity for a user as part of a bind operation.
- entry change notification control - This is a control that is included in search result entry messages performed as part of a persistent search to indicate how an entry has been updated.
- get effective rights control - This is a request control that may be used to obtain information about what rights a user has for accessing a given entry.
- LDAP assertion control - This is a request control that may be used to ensure that an operation is only processed if the target entry matches a given assertion filter.
- LDAP no-op control - This is a request control that may be used to ensure that a write operation does not actually change any information in the server but attempts to determine whether the operation would otherwise be successful.

- LDAP post-read control - This is a pair of request and response controls that may be used to retrieve an entry as it appeared immediately after performing an add, modify, or modify DN operation.
- LDAP pre-read control - This is a pair of request and response controls that may be used to retrieve an entry as it appeared immediately before performing a delete, modify, or modify DN operation.
- manage DSA IT control - This is a request control that may be used to request that the server treat smart referrals as regular entries rather than as referrals.
- matched values control - This is a request control that may be used to request that entries returned from a search operation only include values matching a given filter.
- persistent search control - This is a request control that may be used to receive notification whenever an entry matching a given set of criteria is updated in the server.
- proxied authorization control - This is a request control that may be used to request that an operation be performed under the authorization of another user.
- server-side sort control - This is a request control that may be used to request that the server sort the results before returning them to the client.
- simple paged results control - This is a request control that may be used to request that the server retrieve only a subset of the results, and when used repeatedly can allow the client to page through the result set.
- virtual list view control - This is a pair of request and response controls that may be used to retrieve an arbitrary page of search results from the server.

An LDAP control is defined as follows:

```
Control ::= SEQUENCE {
     controlType             LDAPOID,
.... criticality             BOOLEAN DEFAULT FALSE,
.... controlValue            OCTET STRING OPTIONAL }
```

A control includes these elements:

- An object identifier that specifies the type of control.
- A criticality, which indicates whether the control should be considered a critical part of the operation (that is, if the server cannot process the control, the operation should fail).
- An optional value, which can be used to provide additional information about the way the control should be processed.

A.4.3.14 CRAM-MD5 SASL mechanism

The CRAM-MD5 Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server with a username and password in a manner that does not expose the clear-text password, so it is significantly safer than simple authentication or the PLAIN SASL mechanism when the connection between the client and the server is not secure.

The CRAM-MD5 SASL mechanism is very similar to DIGEST-MD5 SASL mechanism, but it is somewhat weaker because CRAM-MD5 only includes random data from the server whereas DIGEST-MD5 includes random data from both the client and the server. DIGEST-MD5 also provides a provision for ensuring connection integrity, confidentiality, or both that CRAM-MD5 does not offer. See Authenticating Using CRAM-MD5 SASL Mechanism

A.4.3.15 crypt algorithm

The crypt algorithm is a mechanism for encoding user passwords on Linux and UNIX systems. The CRYPT password storage scheme is an umbrella for all modular crypt password encodings and ensures compatibility with existing implementations.

The modular crypt password encoding is in the format `$<ID>$<SALT>$<PWD>` or `$<ID>$rounds=<N>$<SALT>$<PWD>`, which allows multiple algorithms under the same CRYPT password storage scheme. The supported modular algorithms are MD5, SHA256, and SHA512.

The existing UNIX crypt algorithm is still supported and is the default scheme to ensure compatibility with existing deployments.

You can chose the algorithm you want to use when encoding using the CRYPT password storage scheme. Passwords already encoded with supported algorithms will continue to function, regardless of the currently configured algorithm. Optionally, you can also configure the number of rounds for SHA-based algorithms, because existing passwords imported from another system and schemes using a custom number of key stretching rounds are compatible.

Custom length salts are also supported but only for one-way compatibility. That is, existing passwords with custom length salts are supported, but new passwords always use the maximum salt length specified for each algorithm.

See also password storage scheme.

A.4.4 D

A.4.4.1 database

A database is a repository that is used for storing information. In the directory server, databases are used as the mechanism for storing data in a back end. The primary database used by the directory server is the Berkeley DB Java Edition, although it is possible to create other back ends with different backing stores.

A.4.4.2 database cache

The database cache is a portion of memory that is reserved for holding content from the underlying database. Whenever an attempt is made to retrieve information from the database, the database will first check this cache before going to disk. The database cache can help significantly improve performance by avoiding costly disk I/O.

The database cache may be used either instead of or in addition to the server's entry cache. The database cache frequently creates a more compact representation of the data (which means that more data can be held in the cache in systems with limited memory), but the entry cache generally holds data in a format that can be more efficiently used by the server.

A.4.4.3 debug log

The debug log is a mechanism for obtaining information that you can use to debug problems that occur in the server. Debug information is generally data that is useful only if you have a problem, and is frequently too voluminous to maintain under normal operations. The debug log can report information such as the following:

- Detailed information about exceptions thrown within the server
- Information about data read from or written to network clients
- Information about information read from or written to the database
- Information about decisions made in areas like access control or password policy processing

A.4.4.4 delete operation

The LDAP delete operation can be used to remove an entry from the server (or when used with the subtree delete control, a subtree). The delete request protocol op is defined as follows:

```
DelRequest ::= [APPLICATION 10] LDAPDN
```

The request includes only the DN of the entry to delete.

The response to an LDAP delete operation is an LDAP result element as defined below:

```
DelResponse ::= [APPLICATION 11] LDAPResult
```

A.4.4.5 deprecated password storage scheme

A deprecated password storage scheme is a password storage scheme that is available for use in the server, but is intended primarily for transitional use. If a user has a password encoded with a deprecated storage scheme, then the user will be allowed to authenticate but the password will be re-encoded using the set of default storage schemes defined in the password policy.

This mechanism is primarily intended for cases in which data has been migrated into the directory server from another server uses a password storage scheme that you do not want to continue using (for example, because it is weaker than the default schemes). As users authenticate to the server, their passwords will be transitioned from the deprecated schemes to the default schemes.

A.4.4.6 dereference policy

The dereference policy is an element of a search operation that specifies how the server should handle alias entries that may be encountered during search processing. Allowed alias dereference policy values include:

neverDerefAliases
The server should not attempt to dereference any aliases that it encounters during search processing.

derefInSearching
The server should dereference any entries within the scope of the search operation to determine whether they match the search criteria. The entry specified as the search base DN will not be dereferenced.

derefFindingBaseObj
The server should dereference the entry referenced as the search base DN if it is an alias, but any other alias entries within the scope of the search operation will not be dereferenced.

derefAlways
The server will dereference any alias entries within the scope of the search operation and will also dereference the base entry if it is an alias.

A.4.4.7 DIGEST-MD5 SASL mechanism

The DIGEST-MD5 Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server with a username and password in a manner that does not expose the clear-text password, so it is significantly safer than simple authentication or the PLAIN SASL mechanism when the connection between the client and the server is not secure.

The DIGEST-MD5 SASL mechanism is very similar to CRAM-MD5 SASL mechanism, but it is somewhat strong because CRAM-MD5 includes only random data from the server whereas DIGEST-MD5 includes random data from both the client and the server. DIGEST-MD5 also provides a provision for ensuring connection integrity, confidentiality, or both that CRAM-MD5 does not offer. See About DIGEST-MD5 SASL Mechanism

A.4.4.8 directory information tree (DIT)

The directory information tree, or DIT, refers to the hierarchical structure of the data in a Directory Server. The DIT contains one or more naming context, which are the base entries for the server, and every other entry is descended from one of those naming context entries. That is, a naming context entry is special in that it does not have a parent entry.

Consider a scenario, where the entry `dc=example,dc=com` is the naming context, and it has two immediate children, with DNs of `ou=People,dc=example,dc=com` and `ou=Groups,dc=example,dc=com`, respectively, and each of those entries has its own subordinate entries. There is no predefined limit to the maximum depth of a directory tree, and any entry can potentially have one or more subordinate entries. An entry that does not contain any subordinates is said to be a leaf entry, and any entry that has at least one subordinate entry is called a non-leaf entry.

A.4.4.9 directory manager

The term `directory manager` is a common name used to refer to a root DN user in the Directory Server. It is so named because the default root user typically uses a bind distinguished name of `cn=Directory Manager`. Unlike many other types of directory servers, the directory server allows multiple root DNs to be defined, although the default root DN is still `cn=Directory Manager`.

A.4.4.10 directory server

A directory server is a type of network daemon that stores data in a manner accessible to external clients. Directory servers typically use Lightweight Directory Access Protocol or Directory Services Markup Language (DSML) for communicating with clients, although some servers use other protocols like DAP or NDS.

Directory servers store data in a hierarchical form (called the directory information tree (DIT)) and provide the ability for clients to interact with that information, including:

- search operation, which make it possible to find all entry matching a given set of criteria
- add operation, which make it possible to add new entries to the server
- delete operation, which make it possible to remove entries from the server
- modify operation, which make it possible to update existing information in the server
- modify DN operation, which make it possible to rename entries in the server
- bind operation, which make it possible to authenticate users to the server

- compare operation, which make it possible to determine whether entries have a particular attribute-value pair

The directory server uses LDAPv3 for communicating with network clients, and provides a DSML gateway that can be used to handle DSML requests.

A.4.4.11 directory server agent (DSA)

A directory server agent (DSA) is a single instance of a directory server.

A.4.4.12 Directory Services Markup Language (DSML)

The Directory Services Markup Language (DSML) is a protocol that may be used to communicate with directory server. DSML is an alternative to Lightweight Directory Access Protocol, and uses an XML-based representation of requests and responses instead of the Basic Encoding Rules encoding that LDAP uses.

In general, DSML is seen as a relatively weak alternative to LDAP because it provides very little benefit and incurs a significant cost because the XML representation is much more verbose and expensive to process when compared with the BER encoding that LDAP uses. In most cases, it is recommended that LDAP be used instead of DSML to interact with the server.

A.4.4.13 distinguished name

A distinguished name (often referred to as a `DN`) is a string that uniquely identifies an entry in the Directory Server. It consists of zero or more distinguished name (RDN) components that identify the location of the entry in the directory information tree (DIT). An entry's distinguished name can be thought of as a kind of an analog to an absolute path in a filesystem in that it specifies both the name and hierarchical location.

The RDN components for a distinguished name are separated by commas and are ordered from right to left. The rightmost components of a DN are closest to the server's naming context, and the leftmost components are closest to the leaf entry. That is, if you think of a directory hierarchy as a kind of pyramid with the naming context at the top and the branches descending downward, then the order of RDN components in a DN are listed from bottom to top.

Even though a DN consists of a series of RDN components, when one refers to an entry's RDN, then it is a reference to the leftmost RDN component. The attributes contained in an entry's RDN must also be contained in that entry.

In a DIT, the top entry is the naming context and its DN is `dc=example,dc=com`. To conserve space, only the RDNs of the subordinate entries are displayed, but the full DNs can be obtained by appending the RDN components from bottom to top. For example, the DN of the leftmost entry on the bottom row would be `uid=ann,ou=People,dc=example,dc=com`.

See RFC 4514 (`http://www.ietf.org/rfc/rfc4514.txt`) for more information about LDAP distinguished names and the way in which they should be represented as strings.

A.4.4.14 distribution

Distribution is a proxy deployment type in which data is split into *partitions*. The split of data is determined by a distribution algorithm.

A.4.4.15 DIT

See directory information tree (DIT).

A.4.4.16 DIT content rule

A DIT content rule is a schema element that specifies which auxiliary object class are allowed to be used with an entry, as well as which attribute type are required, allowed, and prohibited for use with an entry, based on its structural object class.

The components of a DIT content rule definition include:

- The numeric object identifier of the structural object class with which the DIT content rule is associated.
- An optional set of names for the DIT content rule.
- An optional set of auxiliary object class names or OIDs for the auxiliary classes that are allowed to be used with entries containing the associated structural class.
- An optional set of attribute type names or OIDs for attribute types that are required to be present in entries with the associated structural class. These attributes will be required even if they are not allowed by any of the object classes in the entry.
- An optional set of attribute type names or OIDs for attribute types that may optionally be present in entries with the associated structural class. These attributes will be allowed even if they are not allowed by any of the object classes in the entry.
- An optional set of attribute type names or OIDs for attribute types that are prohibited to be present in entries with the associated structural class. These attributes will be prohibited even if they are allowed by any of the object classes in the entry.

The set of DIT content rules defined in the server may be determined by retrieving the `dITContentRules` attribute of the subschema subentry. For more information about DIT content rules, see Overview of DIT Content Rules.

A.4.4.17 DIT structure rule

A DIT structure rule is a schema element that may be used to define the hierarchical relationships between entries. In particular, it defines the kinds of parent entries (based on their structural object class) that an entry with a given structural class is allowed to have.

The components of a DIT structure rule definition include:

- An integer rule ID value that is used to uniquely identify the rule.
- An optional set of names for the DIT structure rule.
- The name or object identifier of the name form with which the DIT structure rule is associated. The name form in turn links the DIT structure rule to a structural object class.
- An optional set of superior rule IDs. If a set of superior rules is defined, then they are used to define the structural classes below which the structural class associated with the rule's name form is allowed to exist.

The set of DIT structure rules defined in the server may be determined by retrieving the `dITStructureRules` attribute of the subschema subentry. For more information about DIT structure rules, see the Understanding DIT Structure Rules.

A.4.4.18 DN

See distinguished name.

A.4.4.19 DSA

See directory server agent (DSA).

A.4.4.20 DSA-specific entry

A DSA-Specific Entry (DSE) is a special type of entry that provides information about a directory server agent (DSA), which is a synonym for directory server.

Lightweight Directory Access Protocol defines a special entry called the root DSE that provides information about the information contained in the server and the types of operations that it supports.

A.4.4.21 DSE

See DSA-specific entry.

A.4.4.22 DSML

See Directory Services Markup Language (DSML).

A.4.4.23 DSML gateway

A DSML gateway (or DSML-to-LDAP gateway) is a special type of network daemon that is used to translate between Directory Services Markup Language (DSML) and Lightweight Directory Access Protocol. In general, a DSML gateway accepts DSML requests from clients, converts them to LDAP requests that it forwards to a directory server for processing. It then translates the LDAP response from the directory server back to DSML to return to the client.

The directory server supports DSML through a DSML gateway, which is implemented as a Web application that can run in an application server.

A.4.4.24 duration

Certain configuration properties take a duration as their allowed value.

A duration includes an integer, and a unit, specified in weeks (w), days (d), hours (h), minutes (m), seconds (s), or miliseconds (ms), or some combination with multiple specifiers. For example, you can specify one week as `1w`, `7d`, `168h`, `10080m`, or `604800s`. Or you can specify ten and a half days as `1w3d12h0m0s`.

Not all properties that require a duration support all duration specifiers (`w`, `d`, `h`, `m`, `s`, and `ms`).

A duration property can also include the following:

- base unit

  Specifies the minimum granularity that can be used to specify duration property values. For example, if the base unit is in seconds, values represented in milliseconds are not permitted.

- maximum unit (optional)

  Specifies the largest duration unit that can be used to specify duration property values. Values presented in units greater than this unit are not permitted.

- lower limit

Specifies the smallest duration permitted by the property.

- upper limit (optional)

  Specifies the largest duration permitted by the property.

- unlimited duration

  Certain properties allow you to specify an unlimited duration. This is represented using the decoded value, -1, or the encoded string value `unlimited`.

A.4.4.25 dynamic group

A dynamic group is a type of group in the directory server that defines its membership using a set of search criteria in the form of an LDAP URL, as opposed to a static group in which the distinguished name of the members are explicitly specified.

Dynamic groups provide an efficient way to manage groups with very large numbers of members. They are much more scalable than static groups, and their membership is automatically updated as entry change so that the match or no longer match the group criteria.

A.4.5 E

A.4.5.1 entry

An entry is the structure that holds information in a directory server. It consists of the following components:

- A distinguished name that uniquely identifies the entry among all other entries in the server.
- A collection of object class values that are used to govern the contents of the entry.
- A collection of attribute that contain the actual data for the entry.

An entry must always have exactly one structural object class that defines what type of entry it is. It may have zero or more auxiliary object class that may be used identify other characteristics for the entry. Together, the structural and auxiliary classes define a set of required attributes, which must be present in the entry, and optional attributes, which may be included in the entry but are not required.

A.4.5.2 entry cache

The entry cache is a mechanism that uses system memory for holding entries in a manner that may be quickly accessed so that it is not necessary to decode them from the database whenever they are needed. Entry caching mechanisms are particularly effective when used with applications that access the same entry multiple times in a sequence of operations. For example, an application which first search operation to find a user entry and then bind operation as that user to verify a password, which is a very common usage pattern.

The entry cache may be used either instead of or in addition to the server's database cache. The database cache generally uses a more compact representation of the data, but the entry cache generally holds data in a format that can be more efficiently used by the server.

Unlike the database cache, which is maintained by the underlying database, the entry cache is managed by the directory server itself. There are several different entry cache implementations that may be used.

A.4.5.3 entry change notification control

The entry change notification control is a control that is included in search result entries returned to clients in response to a search operation that uses the persistent search control. This control contains additional information about the change made to the entry, including the type of change made, the change number (which corresponds to an item in the server's change log, if the server supports a change log), and, if the entry was renamed, the old DN of the entry. The draft-ietf-ldapext-psearch-03 (http://tools.ietf.org/html/draft-ietf-ldapext-psearch-03) describes this control, which has an OID of 2.16.840.1.113730.3.4.7.

The control is defined as follows:

```
EntryChangeNotification ::= SEQUENCE {
          changeType ENUMERATED {
               add             (1),
               delete          (2),
               modify          (4),
               modDN           (8)
          },
          previousDN   LDAPDN OPTIONAL,     -- modifyDN ops. only
          changeNumber INTEGER OPTIONAL     -- if supported
}
```

A.4.5.4 entryDN

An entryDN is an operational attribute that provides a copy of the entry's current distinguished name. Because a DN is not an attribute of the entry, it cannot be used to perform attribute value assertions. RFC 5020 describes the entryDN that provides a mechanism to access an entry's DN.

A.4.5.5 entry ID

An entry ID is an integer value that is used to uniquely identify an entry in the Directory Server back end. Although the entry's distinguished name could be used for this purpose, the numeric entry ID is much more compact and more efficient to decode, so it is more appropriate for widespread use.

The entry ID is used as the key to the actual entry data in the id2entry database, and it is used in ID list to identify entries matching the associated index key.

A.4.5.6 entryUUID

An entryUUID is a universally unique identifier that is contained in the `entryUUID` operational attribute and is assigned to each entry in the directory server. It is defined in RFC 4530 (http://www.ietf.org/rfc/rfc4530.txt) and it is intended to be a unique identifier that will not change over the life of the entry (as opposed to the distinguished name, which can change as a result of a modify DN operation). Because of the greater stability of the entryUUID, it is used by the replication subsystem to track entries even if the DN does change.

A.4.5.7 equality index

An equality index is a type of indexwhich is used to identify efficiently which entries are exactly equal to a given assertion value. An equality index may only be maintained for attributes that have a corresponding equality matching rule. That matching rule will be used to normalized value to use as index keys, and the value for that key will be the ID list containing the entry ID of the entries with values that are equal to that normalized value.

A.4.5.8 equality search filter

An equality search filter is a type of LDAP search filter that can be used to identify entries that contain a specific value for a given attribute. The server will use an equality matching rule to make the determination.

The string representation of an LDAP equality filter comprises an opening parenthesis followed by the attribute name, an equal sign, the attribute value, and the closing parenthesis. For example, an equality filter of `(uid=john.doe)` will match any entry in which the `uid` attribute contains a value of `john.doe`.

A.4.5.9 error log

The error log provides a mechanism for reporting errors, warnings, and other significant events that happen in the life of the server. Each message written to the error log will include a category (indicating the area of the server in which the message was generated) and severity (indicating the relative importance of the message), along with an integer value that uniquely identifies the associated message string.

A.4.5.10 export

See LDIF export.

A.4.5.11 extended operation

The LDAP extended operation provides a degree of extensibility to the LDAP protocol by allowing clients to request operations not defined in the core protocol specification. Examples of LDAP extended operations include:

cancel extended operation
This operation may be used to cancel a previously-requested operation.

Password Modify extended operation
This operation may be used to change a user password.

StartTLS extended operation
This operation may be used to initiate a secure communication channel over an existing connection.

"Who Am I?" extended operation
This operation may be used to determine the authorization identity associated with the client connection.

The extended request protocol op is defined as follows:

```
ExtendedRequest ::= [APPLICATION 23] SEQUENCE {
     requestName      [0] LDAPOID,
     requestValue     [1] OCTET STRING OPTIONAL }
```

The elements of the extended request include:

- The object identifier that is used to indicate the type of operation to perform.
- An optional value containing additional information to use while processing the request.

The response to an LDAP extended operation is defined as follows:

```
ExtendedResponse ::= [APPLICATION 24] SEQUENCE {
     COMPONENTS OF LDAPResult,
     responseName     [10] LDAPOID OPTIONAL,
     responseValue    [11] OCTET STRING OPTIONAL }
```

The extended response includes these elements:

- The elements of the result object.
- An optional OID used to indicate the type of response.
- An optional encoded value with additional information to include in the response.

A.4.5.12 extensible match index

An extensible match index is a type of index that is used to help accelerate search operation using an extensible match search filter. Index keys are values that have been normalized value using a specified matching rule, and the corresponding ID list contains the entry ID for all entries that match the value according to that matching rule.

A.4.5.13 extensible match search filter

An extensible match search filter is a type of LDAP search filter that can be used to identify matching entries using a specified matching rule.

An extensible matching filter contains the following components:

- The OID of the matching rule to use for the determination. This is an optional element, and if it is not provided then the attribute type must be given and its default equality matching rule will be used.
- The name of the attribute type that will be targeted. If this is not provided, then all attributes contained in the entry will be examined.
- A flag that indicates whether the matching should be performed against the attributes of the entry's distinguished name *and* the attributes contained in the entry.
- An assertion value that should be used as the target for the matching rule.

The string representation of an LDAP extensible match filter comprises the following components in order:

- An opening parenthesis
- The name of the attribute type, or an empty string if none was provided
- The string `:dn` if the `dnAttributes` flag is set, or an empty string if not
- If a matching rule ID is available, then a string composed of a colon followed by that OID, or an empty string if there is no matching rule ID
- The string `:=`
- The string representation of the assertion value
- A closing parenthesis

A.4.5.14 EXTERNAL SASL mechanism

The EXTERNAL Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server using information that is available outside of the communication performed at the LDAP protocol level. The most common use of EXTERNAL authentication (and at present, the only form that the directory server supports) is

for the server to identify the client based on a certificate that the client presented during Secure Sockets Layer or StartTLS extended operation negotiation. The Directory Server will use a certificate mapper to map the client's certificate to a user in the directory, and may optionally perform additional validation (for example, ensuring that the presented certificate actually exists in the user's entry).

A.4.6 F

A.4.6.1 failover algorithm

A load balancing algorithm in which all client requests are sent to a main remote LDAP data source. If the main remote LDAP goes down, the request are forwarded to a secondary remote LDAP server, and so on. This ensures the continuation of the service after failure of one or more remote LDAP servers.

A.4.6.2 false filter

See LDAP false filter.

A.4.7 G

A.4.7.1 generalized time

Generalized time is a form at may be used to represent time stamps, along with time zone information. A generalized time value contains the following components:

- Four digits to signify the year.
- Two digits to signify the month (01 for January, 02 for February,..., 12 for December).
- Two digits to signify the day of the month (01 through 28/29/30/31 depending on the month and whether it's a leap year).
- Two digits to signify the hour of the day (00 for midnight through 23 for 11 pm).
- An optional two digits that specify the minute of the hour (between 00 and 59).
- An optional two digits that specify the second of the minute (between 00 and 59, or 60 for leap seconds). This may only be included if the time stamp value also contains the minute of the hour.
- An optional period followed by one or more digits that specify the fraction of a second. This may only be included if the time stamp value contains minute and second information.
- A time zone indicator. This may be either the capital letter `Z` to indicate that the value is in the UTC time zone, or a plus or minus sign followed by two or four digits that specify the offset from UTC time zone.

An example of a time stamp in a generalized time format is `20070508200557Z`, which specifies a time (in the UTC time zone) of 8:05:57 PM on May 28, 2007. An equivalent value in the United States central daylight savings time (a five hour offset from UTC) would be `20070508150557-0500`.

A.4.7.2 get effective rights control

The get effective rights control is a type of control that can be used to determine the rights that a given user has when interacting with a given entry. The control has an object identifier of 1.3.6.1.4.1.42.2.27.9.5.2 and uses the following definition:

```
GetRightsControl ::= SEQUENCE {
     authzId    authzId
     attributes  SEQUENCE OF AttributeType
}

-- Only the "dn:DN form is supported.
```

For an example of using this control in a search request, see Searching Using the Get Effective Rights Control.

A.4.7.3 global index

In a proxy deployment, the global index maps the data entries to the *distribution partition* where the data is stored. Global indexes map a specific attribute (such as `telephonenumber`). For example, the global index could map `telephonenumber=5551212` to distribution partition 1, while `telephonenumber=4441212` to partition 2.

A.4.7.4 global index catalog

A global index catalog contains one or more *global indexes*. A global index catalog can be used with a distribution deployment, to diminish the need for broadcasts, since the values of some attributes are mapped to the partition in which the entry is held.

A.4.7.5 greater than or equal to search filter

An greater or equal search filter is a type of LDAP search filter that can be used to identify entries that contain a specific value for a given attribute that is greater than or equal to the provided assertion value. The server will use an ordering matching rule to make the determination.

The string representation of an LDAP greater or equal search filter comprises an opening parenthesis followed by the attribute name, a greater than sign, an equal sign, the assertion value, and the closing parenthesis. For example, a greater or equal filter of `(createTimestamp>=20070101000000Z)` will match any entry that has a `createTimestamp` value that is greater than or equal to `20070101000000Z`.

A.4.7.6 group

A group is a special type of entry in the Directory Server that is used to represent a set of users in the server. Groups may be used within the server in several different ways, like access control and virtual attribute, and they may also be used by clients for various purposes.

There are several different types of groups defined in the server, including:

- static group provide an explicit list of members
- dynamic group obtain their membership information from a set of search criteria
- virtual static group appear to be static groups but obtain their membership information from another type of group, like a dynamic group

A.4.7.7 GSSAPI SASL mechanism

The GSSAPI Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server using a Kerberos V5 session. Kerberos is a protocol that is commonly used for single sign-on purposes, and provides the option of using integrity, confidentiality, or both to protect the communication between the client and the server (although the directory server does not at present support GSSAPI for protecting network content but only for authenticating clients).

RFC 4752 (http://www.ietf.org/rfc/rfc4752.txt) describes the GSSAPI SASL mechanism.

A.4.8 I

A.4.8.1 ID list

An ID list is used as the value of a Directory Server index. It contains a set of entry ID for all entries that match the associated index key.

In some cases, an ID list can have a special value that indicates that there are more entries matching the index key than allowed by the index entry limit. In that case, the index key will no longer be maintained.

A.4.8.2 id2entry database

The id2entry database is a type of database that maps an entry ID to the contents of the corresponding entry. The entry ID is used in ID list within index.

A.4.8.3 identity mapper

An identity mapper provides logic that can be used to map an authentication ID or authorization ID value to a corresponding user entry. Identity mappers are used with several Simple Authentication and Security Layer mechanisms, as well as the proxied authorization control and the Password Modify extended operation.

A.4.8.4 idle account lockout

Idle account lockout is a part of the Directory Server password policy that may be used to lock user accounts that remain unused for a significant period of time. It requires that the last login time feature be enabled so that user authentication times will be recorded, and any bind operation by a user that has not authenticated within a specified period of time will be rejected.

If a user's account has been locked due to remaining idle for too long, then it may be unlocked by an administrative password reset.

A.4.8.5 in-core restart

An in-core restart is a process by which the server may be restarted without actually existing the JVM used to run the server. It can be used to apply any change that requires a server restart other than one that requires the modification of a JVM argument. An in-core restart may be faster than stopping and re-starting the server process, and it has the added benefit of

maintaining the JIT cache that has been accumulated from observing processing performed within the JVM.

A.4.8.6 index

An index is a mechanism used by the Directory Server database that can be used to efficiently find entries matching search criteria. An index maps a key to an ID list, which is the set of entry ID for the entries that match that index key.

The directory server uses six primary types of indexes:

- approximate index are used to identify entries containing attribute values approximately equal to a given assertion value.
- equality index are used to identify entries containing an attribute value that exactly matches a given assertion value.
- extensible match index are used to identify entries that match a given extensible match filter. This index is not currently supported.
- ordering index are used to identify entries that have values that are greater than or equal to, or less than or equal to, a given assertion value.
- presence index are used to identify entries that contain at least one value for a given attribute.
- substring index are used to identify entries that contain an attribute value matching a given substring assertion.

A.4.8.7 index entry limit

The index entry limit is a configuration limit that can be used to control the maximum number of entries that is allowed to match any given index key (that is, the maximum size of an ID list). This provides a mechanism for limiting the performance impact for maintaining index keys that match a large percentage of the entries in the server. In cases where large ID lists might be required, performing an unindexed search can often be faster than one that is indexed.

The index entry limit in the directory server is analogous to the ALL IDs threshold in Oracle Directory Server Enterprise Edition.

A.4.8.8 intermediate response

See LDAP intermediate response.

A.4.8.9 Internet Draft

An Internet Draft is a form of specification defined through the IETF (`http://www.ietf.org/`). Internet drafts are short-lived specifications that typically go through multiple revisions, and may change significantly between revisions. Internet Drafts that reach a point of stability may be promoted to request for comments. Other drafts may stagnate and become no longer maintained, although in some cases they may still describe viable functionality that is worth implementing in the server.

A.4.9 J

A.4.9.1 Java Management Extensions

Java Management Extensions (JMX) is a framework is a Java technology that can be used for accessing monitoring and configuration information.

Oracle Unified Directory uses JMX for publishing information from monitor entry. It also uses the JMX notification mechanism for administrative alerts if there are significant problems or events in the server.

A.4.9.2 JMX

See Java Management Extensions.

A.4.10 K

A.4.10.1 key manager provider

A key manager provider is a component of the server that can provide access to private key information for server certificate.

The key manager providers available for use in the server include the following:

- A mechanism for accessing key information in a JKS keystore
- A mechanism for accessing key information in a PKCS#12 file
- A mechanism for accessing key information in a PKCS#11 token

A.4.11 L

A.4.11.1 last login time

The last login time feature of the Directory Server is a mechanism that can be used to write the time that the user last authenticated to the server using a bind operation. The last login time may be written to a specified attribute with a user-defined format.

Be aware that in many servers, it may be desirable to define the last login time format to contain only the date but not the time of day. If this format is used, then the value will be only updated once per day, thereby reducing the potential impact on performance for users that authenticate several times throughout the day.

The last login time may be maintained for informational purposes, but it can also be used to enable the idle account lockout feature.

A.4.11.2 lastmod plug-in

The lastmod plug-in is a pre-operation idle account lockout that can be used to add the `creatorsName` and `createTimestamp` attributes to an entry as part of an add operation, or update the `modifiersName` and `modifyTimestamp` attributes in an entry as part of a modify operation or modify DN operation operation.

A.4.11.3 LDAP assertion control

The LDAP assertion control is a type of control that may be used to perform an operation only if the target entry matches a given assertion filter. It may be used with compare operation, delete operation, modify operation, modify DN operation, and search operation.

RFC 4528 (http://www.ietf.org/rfc/rfc4528.txt) describes the LDAP assertion control, which has an OID of 1.3.6.1.1.12. The value of the control should be encoded as an LDAP LDAP search filter.

For an example of using this control in a search request, see Searching Using the LDAP Assertion Control..

A.4.11.4 ldapcompare command

The ldapcompare command can be used to request an LDAP compare operation.

For information about using this command, see ldapcompare.

A.4.11.5 LDAP Data Interchange Format

The LDAP Data Interchange Format (LDIF) is a mechanism form representing directory data in text form. The LDIF specification is contained in RFC 2849 (http://www.ietf.org/rfc/rfc2849.txt) and describes a format not only for representing directory data but also a mechanism for making changes to that data.

In general, an LDIF record consists of a series of name-value pairs. The name can be followed by a single colon, zero or more spaces, and associated value, or it can be followed by two colons, zero or more spaces, and the base64 encoding representation of the value. Each name-value pair is given on a separate line, and long lines may be wrapped onto two or more lines using an end-of-line character followed by exactly one space at the beginning of the next line. LDIF records should be separated from each other by at least one blank line. Any line that begins with an octothorpe (#) character will be treated as a comment and ignored.

For an LDIF representation of an entry, the first line should contain the distinguished name of the entry. The remaining lines of the LDIF record will represent the attribute of the entry, with the attribute description used as the name. Multivalued attributes will be represented with a separate line per value.

The following provides an example of a user entry represented in the LDAP Data Interchange Format:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: john.doe
givenName: John
sn: Doe
cn: John Doe
mail: john.doe@example.com
userCertificate;binary:: MIIB5TCCAU6gAwIBAgIERloIajANBgkqhkiG9w0BAQUFADA3M
 QswCQYDVQQGEwJVUzEVMBMGA1UEChMMRXhhbXBsZSBDb3JwMREwDwYDVQQDEwhKb2huIERvZT
 AeFw0wNzA1MjcyMjM4MzRaFw0wNzA4MjUyMjM4MzRaMDcxCzAJBgNVBAYTAlVTMRUwEwYDVQQ
 KEwxFeGFtcGxlIENvcnAxETAPBgNVBAMTCEpvaG4gRG9lMIGfMA0GCSqGSIb3DQEBAQUAA4GN
 ADCBiQKBgQCWNZB4qs1UvjYgvGvB9udmiUi4X4DeaSm3o0p8PSwpOFxSqgWdSwKgUugZ1EJVy
 YoakljDFsJ0GVown+dIB24V4ozNs6wa0YotIKTV2AcySQkmzzP3e+OnE9Aa1wlB/PVnh1CFLg
```

```
 klUOoruLE10bac5HA8QiAmfNMorU26AwFTcwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAGrzMKN
 bBRWn+LIfYTfqKYUc258XVbhFri1OV0oF82vyvciYWZzyxLc52EPDsymLmcDh+CdWxy3bVkjd
 Mg1WEtMGr1GsxOVi/vWe+kT4tPhinnB4Fowf8zgqiUKo9/FJN26y7Fpvy1IODiBInDrKZRvNf
 qemCf7o3+Cp00OmF5ey
userPassword: {SSHA}s4Bd9M0tCpRDr8/U+IXetRcAbd8bJY3AFKsn+A==
```

To represent an LDAP add operation in LDIF, the format is exactly the same as the format used to represent an entry, except that the line immediately after the DN should indicate a `changetype` of `add`, as shown in the following example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: john.doe
givenName: John
sn: Doe
cn: John Doe
mail: john.doe@example.com
userCertificate;binary:: MIIB5TCCAU6gAwIBAgIERloIajANBgkqhkiG9w0BAQUFADA3M
 QswCQYDVQQGEwJVUzEVMBMGA1UEChMMRXhhbXBsZSBDb3JwMREwDwYDVQQDEwhKb2huIERvZT
 AeFw0wNzA1MjcyMjM4MzRaFw0wNzA4MjUyMjM4MzRaMDcxCzAJBgNVBAYTAlVTMRUwEwYDVQQ
 KEwxFeGFtcGxlIENvcnAxETAPBgNVBAMTCEpvaG4gRG9lMIGfMA0GCSqGSIb3DQEBAQUAA4GN
 ADCBiQKBgQCWNZB4qs1UvjYgvGvB9udmiUi4X4DeaSm3o0p8PSwpOFxSqgWdSwKgUugZ1EJVy
 YoakljDFsJ0GVown+dIB24V4ozNs6wa0YotIKTV2AcySQkmzzP3e+OnE9Aa1wlB/PVnh1CFLg
 klUOoruLE10bac5HA8QiAmfNMorU26AwFTcwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAGrzMKN
 bBRWn+LIfYTfqKYUc258XVbhFri1OV0oF82vyvciYWZzyxLc52EPDsymLmcDh+CdWxy3bVkjd
 Mg1WEtMGr1GsxOVi/vWe+kT4tPhinnB4Fowf8zgqiUKo9/FJN26y7Fpvy1IODiBInDrKZRvNf
 qemCf7o3+Cp00OmF5ey
userPassword: password
```

To represent an LDAP delete operation in LDIF, the format is simply a line containing the DN of the entry followed by a line indicating a `changetype` of `delete`, like:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: delete
```

To represent an LDAP modify operation in LDIF, the format is a little more complex. The first line should contain the DN of the entry, and the second should contain a `changetype` of `modify`. The third line should specify the attribute modification type (add, delete, replace, or increment) followed by the attribute description, and there may be additional lines that specify specific values for that change, with the name portion being the attribute description and the value being the corresponding attribute value. There may be multiple attribute modifications described in a single modify change record, with each of them separated by a line containing only a dash, as shown in the following example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: modify
replace: userPassword
userPassword: newpassword
-
replace: description
description: This is the first description value
description: This is the second description value
```

To represent an LDAP modify DN operation in LDIF, the first line should contain the DN of the entry, and the second line should contain a `changetype` of `moddn`. The third line should have a name of `newrdn` with a value equal to the new RDN to assign to the entry, and the fourth should

have a name of `deleteoldrdn` followed by a value of either 1 (if the `deleteOldRDN` flag should be `true`) or `0` (if it should be false). There can be an optional fifth line with a name of `newsuperior` and a value of the new superior DN if one is included in the request. For example:

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: moddn
newrdn: uid=johnathan.doe
deleteoldrdn: 1
```

A.4.11.6 ldapdelete command

The ldapdelete command can be used to request an LDAP delete operation.

For information about using this command, see ldapdelete.

A.4.11.7 LDAP false filter

An LDAP false filter is a special type of OR search filter that does not contain any embedded filter components. It is called an "LDAP false filter" because it always evaluates to `false` and will never match any entry.

The string representation for an LDAP true filter is `(|)`. LDAP false filters are described in RFC 4526 (`http://www.ietf.org/rfc/rfc4526.txt`).

A.4.11.8 LDAP intermediate response

The LDAP intermediate response message is a special type of protocol op that allows the server to send additional messages providing information about the state of an operation before it has completed processing and the final response message is sent. Prior to the introduction of the intermediate response in RFC 3771 (`http://www.ietf.org/rfc/rfc3771.txt`), only search operations were allowed to send multiple responses.

The intermediate response protocol op is defined as follows:

```
IntermediateResponse ::= [APPLICATION 25] SEQUENCE {
          responseName     [0] LDAPOID OPTIONAL,
          responseValue    [1] OCTET STRING OPTIONAL }
```

At present, the directory server does not support any operations that use intermediate response messages.

A.4.11.9 LDAP message

The LDAP message is the fundamental protocol data unit for LDAP communication. It is the container that is used to hold all request and response elements.

The LDAP message is defined as shown in the following example:

```
LDAPMessage ::= SEQUENCE {
     messageID       MessageID,
     protocolOp      CHOICE {
          bindRequest           BindRequest,
          bindResponse          BindResponse,
          unbindRequest         UnbindRequest,
          searchRequest         SearchRequest,
          searchResEntry        SearchResultEntry,
          searchResDone         SearchResultDone,
          searchResRef          SearchResultReference,
```

```
        modifyRequest         ModifyRequest,
        modifyResponse        ModifyResponse,
        addRequest            AddRequest,
        addResponse           AddResponse,
        delRequest            DelRequest,
        delResponse           DelResponse,
        modDNRequest          ModifyDNRequest,
        modDNResponse         ModifyDNResponse,
        compareRequest        CompareRequest,
        compareResponse       CompareResponse,
        abandonRequest        AbandonRequest,
        extendedReq           ExtendedRequest,
        extendedResp          ExtendedResponse,
        ...,
        intermediateResponse  IntermediateResponse },
    controls       [0] Controls OPTIONAL }
```

The LDAP message includes these elements:

- The message ID, which is the unique identifier that is used to correlate requests and responses. The client includes a message ID in the request, and all response messages for that request will have the same message ID.
- The protocol op, which is the container for the actual request or response.
- An optional set of control that can be used to provide additional information about the way that the request should be processed, or additional information about the response from the server.

A.4.11.10 LDAP modify DN operation

You can use the LDAP modify DN operation to change the distinguished name of an entry in the Directory Server. This operation can alter the relative distinguished name of the entry, it can move the entry below a new parent, or it can do both. If the target entry has subordinate entries, then you can use it to move or rename that subtree.

The modify DN request protocol op is defined as follows:

```
ModifyDNRequest ::= [APPLICATION 12] SEQUENCE {
     entry           LDAPDN,
     newrdn          RelativeLDAPDN,
     deleteoldrdn    BOOLEAN,
     newSuperior     [0] LDAPDN OPTIONAL }
```

The modify DN request includes these elements:

- The DN of the entry to rename, move, or rename and move.
- The new RDN to use for the entry. If the entry is simply to be moved below a new parent, then it may be the same as the current RDN.
- A flag that indicates whether the current RDN attribute values should be removed from the entry.
- An optional DN specifying the new parent for the entry.

The response to an LDAP modify DN operation is an LDAP result as defined as follows:

```
ModifyDNResponse ::= [APPLICATION 13] LDAPResult}
```

A.4.11.11 LDAP modify operation

The LDAP modify operation can be used to alter an existing entry in the Directory Server. The modify request protocol op is defined as follows:

```
ModifyRequest ::= [APPLICATION 6] SEQUENCE {
     object          LDAPDN,
     changes         SEQUENCE OF change SEQUENCE {
          operation       ENUMERATED {
               add     (0),
               delete  (1),
               replace (2),
               ...  },
          modification    PartialAttribute } }
```

The modify request includes these elements:

- The DN of the entry to modify
- One or more modification elements indicating the changes to make in the entry

The response to an LDAP modify operation is an LDAP result defined as shown here:

```
ModifyResponse ::= [APPLICATION 7] LDAPResult
```

A.4.11.12 ldapmodify command

The ldapmodify command may be used to request LDAP add operation, delete operation, modify operation, and modify DN operation operations.

For information about using this command, see re.

A.4.11.13 LDAP no-op control

The LDAP no-op control is a type of control that may be attached to an LDAP add operation, delete operation, modify operation, or modify DN operation to indicate that it should not actually make any change to the content in the server.

The LDAP no-op control is defined in draft-zeilenga-ldap-noop. This is a specification that is still in progress, but the directory server does provide basic support for this control using an object identifier of 1.3.6.1.4.1.4203.1.10.2. The control does not have a value.

The following example shows the use of the no-op control in an `ldapmodify` operation.

```
ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
-J 1.3.6.1.4.1.4203.1.10.2
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: modify
replace: telephoneNumber
telephoneNumber: +1 995 589 3333

Processing MODIFY request for uid=aaltay,ou=People,dc=example,dc=com
MODIFY operation failed
Result Code:  16654 (No Operation)
Additional Information:  The modify operation was not actually performed in the
Directory Server back end because the LDAP no-op control was present in the request
```

A.4.11.14 LDAP post-read control

The LDAP post-read control is a type of control that may be attached to an LDAP add operation, modify operation, or modify DN operation operation to request that the server return a copy of the target entry exactly as it was at the end of the processing for that operation. It is one of the LDAP read entry controls defined in RFC 4527 (http://www.ietf.org/rfc/rfc4527.txt).

The post-read request control has an OID of 1.3.6.1.1.13.2, and the value should be encoded in the same way as the search attributes in a search operation. The response control has an OID of 1.3.6.1.1.13.2 (the same as the OID for the request control), and the value should be encoded in the same was as a search result entry.

The following example shows the use of the post-read control in an `ldapmodify` request:

```
$ ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
--postReadAttributes=telephoneNumber
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: modify
replace: telephoneNumber
telephoneNumber: +1 995 589 3333

Processing MODIFY request for uid=aaltay,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
Target entry after the operation:
dn: uid=aaltay,ou=People,dc=example,dc=com
telephoneNumber: +1 995 589 3333
```

A.4.11.15 LDAP pre-read control

The LDAP pre-read control is a type of control that may be attached to an LDAP delete operation, modify operation, or modify DN operation operation to request that the server return a copy of the target entry exactly as it was immediately before the processing for that operation. It is one of the LDAP read entry controls defined in RFC 4527 (http://www.ietf.org/rfc/rfc4527.txt).

The pre-read request control has an OID of 1.3.6.1.1.13.1, and the value should be encoded in the same way as the search attributes in a search operation. The response control has an OID of 1.3.6.1.1.13.1 (the same as the OID for the request control), and the value should be encoded in the same was as a search result entry.

The following example shows the use of the pre-read control in an `ldapmodify` request:

```
$ ldapmodify -h localhost -p 1389 -D "cn=directory manager" -j pwd-file \
--preReadAttributes=telephoneNumber
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: modify
replace: telephoneNumber
telephoneNumber: +1 995 589 4444

Processing MODIFY request for uid=user.199,ou=People,dc=exampele,dc=com
MODIFY operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
Target entry before the operation:
dn: uid=aaltay.199,ou=People,dc=example,dc=com
telephoneNumber: +1 995 589 3333
```

A.4.11.16 LDAP result

The LDAP result element is a generic protocol op that is used for the responses of several types of LDAP operations. The basic definition for the LDAP result is as follows:

```
LDAPResult ::= SEQUENCE {
     resultCode         ENUMERATED {
          success                      (0),
          operationsError              (1),
          protocolError                (2),
          timeLimitExceeded            (3),
          sizeLimitExceeded            (4),
          compareFalse                 (5),
          compareTrue                  (6),
          authMethodNotSupported       (7),
          strongerAuthRequired         (8),
               -- 9 reserved --
          referral                     (10),
          adminLimitExceeded           (11),
          unavailableCriticalExtension (12),
          confidentialityRequired      (13),
          saslBindInProgress           (14),
          noSuchAttribute              (16),
          undefinedAttributeType       (17),
          inappropriateMatching        (18),
          constraintViolation          (19),
          attributeOrValueExists       (20),
          invalidAttributeSyntax       (21),
               -- 22-31 unused --
          noSuchObject                 (32),
          aliasProblem                 (33),
          invalidDNSyntax              (34),
               -- 35 reserved for undefined isLeaf --
          aliasDereferencingProblem    (36),
               -- 37-47 unused --
          inappropriateAuthentication  (48),
          invalidCredentials           (49),
          insufficientAccessRights     (50),
          busy                         (51),
          unavailable                  (52),
          unwillingToPerform           (53),
          loopDetect                   (54),
               -- 55-63 unused --
          namingViolation              (64),
          objectClassViolation         (65),
          notAllowedOnNonLeaf          (66),
          notAllowedOnRDN              (67),
          entryAlreadyExists           (68),
          objectClassModsProhibited    (69),
               -- 70 reserved for CLDAP --
          affectsMultipleDSAs          (71),
               -- 72-79 unused --
          other                        (80), ...  },
     matchedDN          LDAPDN,
     diagnosticMessage  LDAPString,
     referral           [3] Referral OPTIONAL }
```

The elements of the LDAP result are:

result code
An integer value that provides generic information about the result of the operation. The definition above specifies several result codes, but several other values are defined in other specifications.

matched DN
A DN value that may specify the DN of the closest superior entry found if the request specified an entry that did not exist. It may be an empty DN if the matched DN element is not appropriate for the response.

diagnostic message
A human-readable message that provides additional information about the result of the processing. It is typically used for error messages, but it may also be present in successful operations. It may be an empty string if there is no message.

referral
A set of LDAP URLs to other servers in which the client may attempt the operation. This element may be absent if there are no referrals.

A.4.11.17 LDAPS

LDAPS is a term that is used to refer to Lightweight Directory Access Protocol communication over Secure Sockets Layer.

A.4.11.18 LDAP search filter

A search filter provides a mechanism for defining the criteria for defining matching entries in an LDAP search operation. There are ten different types of search filters defined in LDAP:

AND search filter
Serve as a container for holding zero or more search filter elements. All search filters contained in the AND filter must match the target entry for the AND filter to match.

OR search filter
Serve as a container for holding zero or more search filter elements. At least one of the search filters contained in the OR filter must match the target entry for the OR filter to match.

NOT search filter
Serves as a container for exactly one search filter element. The embedded filter must not match the target entry for the NOT filter to match.

equality search filter
Provides a mechanism for identifying entries that contain a specified value for a given attribute.

substring search filter
Provides a mechanism for identifying entries with attribute values matching a specified substring.

greater than or equal to search filter
Provides a mechanism for identifying entries with attribute values greater than or equal to a specific value.

less than or equal to search filter
Provides a mechanism for identifying entries with attribute values less than or equal to a specific value.

presence search filter
Provides a mechanism for identifying entries that contain at least one value for a specified attribute.

approximate search filter
Provides a mechanism for identifying entries with attribute values that are approximately equal to a given value.

extensible match search filter
Provides a mechanism for using a matching rule to identify matching entries using an extensible mechanism.

See RFC 4515 (`http://www.ietf.org/rfc/rfc4515.txt`) for more information about LDAP search filters and a mechanism for representing them as strings.

A.4.11.19 ldapsearch command

The `ldapsearch` command can be used to request an LDAP search operation.

For information about using this command, see ldapsearch.

A.4.11.20 LDAP true filter

An LDAP true filter is a special type of AND search filter that does not contain any embedded filter components. It is called an "LDAP true filter" because it always evaluates to `true` and will match any entry.

The string representation for an LDAP true filter is `(&)`. LDAP true filters are described in RFC 4526 (`http://www.ietf.org/rfc/rfc4526.txt`).

A.4.11.21 LDAP Subentry

An LDAP subentry is a type of entry that contains the `ldapSubEntry` object class. These entries are meant to hold operational data for the server. They are kind of like operational attribute in that they are not returned to clients unless explicitly requested by including a request control with an OID of 1.3.6.1.4.1.7628.5.101.1 and no value. This behavior is described in draft-ietf-ldup-subentry.

For an example of using this control in a search, see Searching Using the LDAP Subentry Control..

A.4.11.22 LDAP URL

An LDAP URL is a type of URL that may be used to reference an entry or set of search criteria. The format of an LDAP URL is described in RFC 4516 (`http://www.ietf.org/rfc/rfc4516.txt`) and may include the following elements:

- The address of the directory server
- The port number of the directory server
- The search base DN
- A set of search attributes
- The search scope for the search
- A LDAP search filter for identifying the entries to match

- A set of extensions that provide information about the way in which the search should be processed

All of these elements are optional. Technically, all that is required of an LDAP URL is the string `ldap://`. However, a more complete URL might be `ldap://directory.example.com:389/dc=example,dc=com?cn,givenName,sn?sub?(uid=john.doe)`.

A.4.11.23 LDIF export

An LDIF export operation is a process by which all or part of the content in a Directory Server back end is written to a file using the LDAP Data Interchange Format. An LDIF export can be initiated using the export-ldif command or an LDIF export task.

A.4.11.24 LDIF import

An LDIF import operation is a process by which data can be added to a Directory Server back end from a file with information in the LDAP Data Interchange Format. An LDIF import provides a significantly more efficient means of adding a large number of entries to the server than LDAP add operation.

An LDIF import operation can be initiated using the import-ldif command or with the LDIF import task.

A.4.11.25 leaf entry

A leaf entry is an entry that does not have any subordinate entries in the server.

A.4.11.26 less than or equal to search filter

An less or equal search filter is a type of LDAP search filter that can be used to identify entries that contain a specific value for a given attribute that is less than or equal to the provided assertion value. The server will use an ordering matching rule to make the determination.

The string representation of an LDAP less or equal search filter is composed of an opening parenthesis followed by the attribute name, a less than sign, an equal sign, the assertion value, and the closing parenthesis. For example, a less or equal filter of `(createTimestamp<=20070101000000Z)` will match any entry that has a `createTimestamp` value that is less than or equal to `20070101000000Z`.

A.4.11.27 lexico algorithm

A proxy distribution algorithm, in which the data is split into partitions based on alphabetical delimitations. For example, [A-E[for one partition and [E-H[for the next partition.

A.4.11.28 Lightweight Directory Access Protocol

The Lightweight Directory Access Protocol (LDAP) is a protocol that may be used to communicate with a directory server. It is an open standard that uses the Basic Encoding Rules subset of Abstract Syntax Notation One to encode communication into message.

The core LDAPv3 specification is in RFC 4510 (`http://www.ietf.org/rfc/rfc4510.txt`), with RFC 4511 (`http://www.ietf.org/rfc/rfc4511.txt`) defining the actual encoding for the protocol. A number of other specifications are defined in several request for comments and Internet Draft.

LDAP defines many different types of operations, including:

abandon operation
Provides a way to terminate the processing for an operation in progress

add operation
Provides a way to add a new entry to the server

bind operation
Provides a way to authentication to the server

compare operation
Provides a way to determine whether an entry has a specified attribute value assertion

delete operation
Provides a way to remove entries from the server

extended operation
Provides a way to perform custom processing implemented as an extension to the core LDAP protocol

modify operation
Provides a way to alter the contents of an entry in the server

modify DN operation
Provides a way to rename an entry in the server

search operation
Provides a way to identify all entries that match a given set of criteria

unbind operation
Provides a way to indicate that the client wishes to disconnect from the server

A.4.11.29 load balancing

Load balancing is a proxy deployment type which provides single access to a set of replicated remote LDAP servers. The choice of the remote LDAP server to which a client requests is sent is determined by a load balancing algorithm.

A.4.11.30 lookthrough limit

The lookthrough limit is a configuration option within the Directory Server that can be used to enforce a limit on the number of entries that the server will examine in the course of processing a search operation. This limit applies to all entries that the server examines, regardless of whether it matches the provided search criteria.

The lookthrough limit configuration attribute can be used to limit the impact of unindexed search, or searches with a very large candidate list.

For information about configuring the lookthrough limit, see Assigning Resource Limits on a User Account. and Setting Root User Resource Limits.

A.4.12 M

A.4.12.1 MakeLDIF command

The MakeLDIF command provides a mechanism for generating entry in LDAP Data Interchange Format form. The entries will be generated based on a template containing several tags that can be used to control the way that the data is generated.

For information about using this command, see make-ldif. About Creating MakeLDIF Template Files describes the valid structure and content for MakeLDIF template files.

A.4.12.2 manage DSA IT control

The Manage DSA IT control is a type of control that can be used to request that the server treat smart referral as regular entries. It can be attached to a delete operation, modify operation, or modify DN operation operation to request that the server apply the operation to the entry containing the smart referral rather than sending the referral back to the client. It may also be attached to a search operation to indicate that the server should return the entries containing the smart referrals as search result entry rather than search result reference.

The Manage DSA IT control is defined in RFC 3296 (`http://www.ietf.org/rfc/rfc3296.txt`). It has an object identifier of `2.16.840.1.113730.3.4.2` with no value.

For an example of using this control in a search request, see Searching Using the Manage DSA IT Control.

A.4.12.3 matched DN

A matched DN is an element of an LDAP result object that can provide additional information about the closest matching entry found in the server. It is generally used when a request targets an entry that does not exist, in which case the matched DN should contain the distinguished name of an entry that does exist in the server and is the closest ancestor of the specified entry. For example, if an operation targeted an entry `uid=doesnt.exist,ou=People,dc=example,dc=com` that did not exist but the entry `ou=People,dc=example,dc=com` does exist in the server, then that may be returned as the matched DN.

There is no guarantee that a matched DN is returned from an operation targeting an entry that does not exist, in which case the matched DN element of the LDAP result will be an empty string. This may be used, for example, if the request targeted an entry that does not have any hierarchical relationship with any other entry in the server.

A.4.12.4 matched values control

The matched values control is a type of control that can be attached to a search operation to indicate that only values matching a specified filter should be included in entries returned to the client. It is described in RFC 3876 (`http://www.ietf.org/rfc/rfc3876.txt`).

The request control should have an OID of 1.2.826.0.1.3344810.2.3. The value should be encoded as follows:

```
ValuesReturnFilter ::= SEQUENCE OF SimpleFilterItem

SimpleFilterItem ::= CHOICE {
     equalityMatch   [3] AttributeValueAssertion,
     substrings      [4] SubstringFilter,
     greaterOrEqual  [5] AttributeValueAssertion,
     lessOrEqual     [6] AttributeValueAssertion,
     present         [7] AttributeDescription,
```

```
                approxMatch     [8] AttributeValueAssertion,
                extensibleMatch [9] SimpleMatchingAssertion }

          SimpleMatchingAssertion ::= SEQUENCE {
                matchingRule    [1] MatchingRuleId OPTIONAL,
                type            [2] AttributeDescription OPTIONAL,
--- at least one of the above must be present
                matchValue      [3] AssertionValue}
```

There is no corresponding response control.

For an example of using this control in a search request, see Searching Using the Matched Values Filter Control.

A.4.12.5 matching rule

A matching rule is a schema element that defines how the server should interact with values of an attribute. There are three standard types of matching rules:

- Equality matching rules are used to determine whether one attribute value is equal to another. This determination is generally made based on the normalized value, and ignores insignificant differences (for example, differences in capitalization or extra spaces).
- Ordering matching rules are used to determine the relative order between two values in a sorted list. This is used when performing server-side sort control, but it is also used for greater than or equal to search filter and less than or equal to search filter filter components.
- Substring matching rules are used to determine whether a value contains a given substring search filter.

In addition to these standard matching rules, the directory server defines a fourth type, approximate matching rules, which are used to determine whether one value is approximately equal to another. The definition of "approximately equal to" can vary, but one common use is "sounds like".

Common examples of matching rules include:

- booleanMatch

  An equality matching rule that determines whether two Boolean values are equal to each other.

- caseExactMatch

  An equality matching rule that determines whether two string values are equal to each other, without ignoring differences in capitalization.

- caseExactOrderingMatch

  An ordering matching rule that is used to determine the relative order between two string values, without ignoring differences in capitalization.

- caseExactSubstringsMatch

  A substring matching rule that is used to determine whether a string value contains a given substring, without ignoring differences in capitalization.

- caseIgnoreMatch

  An equality matching rule that determines whether two string values are equal to each other, ignoring differences in capitalization.

- caseIgnoreOrderingMatch

An ordering matching rule that is used to determine the relative order between two string values, ignoring differences in capitalization.

- caseIgnoreSubstringsMatch

  A substring matching rule that is used to determine whether a string value contains a given substring, ignoring differences in capitalization.

- distinguishedNameMatch

  An equality matching rule that determines whether two distinguished name are equal to each other, ignoring extra spaces around commas separating RDN components and equal signs separating RDN names from values. The individual RDN values will be compared based on the matching rules associated with the corresponding RDN attributes.

- generalizedTimeMatch

  An equality matching rule that determines whether two generalized time values are equal to each other.

- generalizedTimeOrderingMatch

  An ordering matching rule that is used to determine the relative order between two generalized time values.

- integerMatch

  An equality matching rule that determines whether two integer values are equal to each other.

- integerOrderingMatch

  An ordering matching rule that is used to determine the relative order between two integer values.

- octetStringMatch

  An equality matching rule that determines whether two values are exactly equal to each other using a byte-for-byte comparison.

In most cases, the directory server will use matching rules in a completely "behind the scenes" manner without the client needing to know about it. Whenever the client references a given attribute type, then the server will automatically know to use the appropriate matching rules for that attribute. However, it is also possible for the client to request that the server use a specific matching rule when performing an operation using an extensible match search filter.

The set of matching rules defined in the server may be determined by retrieving the `matchingRules` attribute of the subschema subentry. For more information about matching rules, see Overview of Matching Rules.

A.4.12.6 matching rule use

A matching rule use is a schema element that can be used to determine which attribute type can be used with a given matching rule. Be aware that this only applies when using extensible match search filter.

A matching rule use definition includes an object identifier for the matching rule that it applies to and a list of the names or OIDs of the attribute types that may be used with that matching rule. If an attribute is not included in this list, then it cannot be used with the associated matching rule. If there is no matching rule use defined for a given matching rule, then it should be assumed that the matching rule can be used with any attribute type.

The set of matching rule uses defined in the server may be determined by retrieving the `matchingRuleUse` attribute of the subschema subentry. For more information about matching rule uses, see Understanding Matching Rule Uses.

A.4.12.7 MD5

MD5 is a one-way message digest algorithm defined in RFC 1321 (`http://www.ietf.org/rfc/rfc1321.txt`). It can be used to encode a value of an arbitrary length into a 128-bit value that cannot be reversed to determine the original cleartext. It is commonly used as a mechanism for checksumming data, and it is also commonly used for encoding passwords and other sensitive information.

Be aware that recent advances in cryptography have discovered weaknesses in the MD5 algorithm. These discoveries do not directly impact the security of the way that the MD5 algorithm is used by the directory server, but nevertheless it may be wise to use a stronger mechanism like the Secure Hash Algorithm.

A.4.12.8 message

See LDAP message.

A.4.12.9 message ID

The message ID is an integer value that is contained in the message and is used to correlate request and response messages. The client chooses a message ID value to include in the request message, and the server will use the same message ID in all response messages. This makes it possible for the client to have multiple requests in progress on the same connection at any given time. All requests in progress at any given time must have different message IDs. The client will typically keep a sequentially-increasing counter for all request messages so that each request gets a different message ID than the last.

Be aware that unsolicited notification messages will always have a message ID value of zero. All other LDAP messages should have a message ID value between 1 and 2147483647.

A.4.12.10 modification

A modification is an element of an LDAP modify operation that describes a change to a single attribute. A modify request may include one or more modifications to the target entry.

A modification consists of a modification type that describes the type of change (add, delete, replace, or increment), and the attribute including the attribute description and zero or more attribute value.

A.4.12.11 modification type

A modification type describes one of the four ways in which an attribute can have its attribute value altered in a modification. The defined modification types are:

- add

  One or more values are to be added to the target attribute. If the attribute does not exist in the target entry, then it will be added with the given values; otherwise the provided values will be appended to the set of values already defined for that attribute. An add modification type must always supply at least one value.

- delete

One or more values are to be removed from the target attribute, or that attribute is to be removed entirely from the target entry. If one or more specific values are given, then only those values are to be removed from the target attribute (and if they represent the entire set of values for that attribute, then that attribute will be removed from the entry). If no values are given, then the entire attribute (regardless of the number of values it contains) is to be removed from the entry.

- replace

  The set of values for the target attribute should be replaced with the given set of values. A replace can have zero or more values, and the behavior is as follows:

  – If the target attribute already exists in the entry with one or more values, and the replace modification does not have any of its own values, then the target attribute will be removed from the entry.

  – If the target attribute already exists in the entry with one or more values, and the replace modification has one or more of its own values, then the existing set of values will be replaced with the new set of values.

  – If the target attribute does not exist in the entry and the replace modification does not have any of its own values, then no action will be taken.

  – If the target attribute does not exist in the entry and the replace modification has one or more of its own values, then the attribute will be created in the entry with the specified set of values.

- increment

  The value of the target attribute should be incremented by the specified amount. The target attribute must exist in the entry with exactly one value, and that value must be an integer. The increment modification must also include exactly one value and that value must be an integer. The existing value is to be incremented by an amount specified by the increment value. If the increment value is negative, then the existing value will be deprecated by an amount equal to the absolute value of the increment value.

A.4.12.12 modify DN operation

See LDAP modify DN operation.

A.4.12.13 modify operation

See LDAP modify operation.

A.4.12.14 monitor entry

A monitor entry is a type of entry in the server that provides information about a server component. It may provide statistical information for performance monitoring, information about the health of the server, or other information that could be of value.

The directory server provides a general-purpose monitor entry with a distinguished name of `cn=monitor`. A number of other monitor entries exist below that point, including:

- Information about each back end configured in the server
- Information about each connection handler configured in the server
- General information about the system on which the server is running
- Information about the state of the server work queue
- Version information for the server

- A stack trace of all threads currently active in the server

A.4.13 N

A.4.13.1 name form

A name form is a schema element that may be used to control which attribute type may be used in the relative distinguished name for an entry based on its structural object class.

A name form definition include these components:

- An object identifier used to uniquely identify the name form.
- A set of zero or more names that can be used to more easily reference the name form.
- The name or OID of the structural object class with which the name form is associated. Any entry with that structural class will be required to have an RDN which conforms to the requirements of the name form.
- An set of one or more attribute type names or OIDs for attributes that must be present in the RDN of entries with the associated structural class.
- An optional set of one or more attribute type names or OIDs for attributes that may optionally be present in the RDN of entries with the associated structural class.

The set of name forms defined in the server may be determined by retrieving the `nameForms` attribute of the subschema subentry. For more information about name forms, see the Understanding Name Forms.

A.4.13.2 naming context

A naming context, also called a suffix, is a top-level entry in the server's directory information tree (DIT). It is an entry that does not have a parent.

The set of naming contexts defined in the server is listed in the `namingContexts` attribute of the root DSE. Naming contexts are visible through workflows.

A.4.13.3 network group

A network group contains a set of criteria that define categories of client connection. If the client request that is sent to the server meets the policies that are attached to the network group, the network group forwards the request to a *workflow*.

A.4.13.4 non-leaf entry

A non-leaf entry is an entry that has at least one subordinate entry in the server.

A.4.13.5 normalized value

A normalized value is a value that has been processed in a way that makes it possible to be efficiently compared against other values. The normalization process is performed using matching rule and varies based on the type of matching rule. Some kinds of transformations that may be made include:

- Converting all characters to lowercase (or uppercase) to eliminate insignificant differences in capitalization

- Eliminating unnecessary spaces in the value
- Converting values which may have multiple representations into a common form

A.4.13.6 notice of disconnection unsolicited notification

The notice of disconnection is a type of unsolicited notification that can be used to indicate that the server is about to close the connection to the client for some reason (for example, the server is being shut down, or the client has remained idle for too long).

The OID for the extended response containing the notice of disconnection is `1.3.6.1.4.1.1466.20036`. It will not have a response value, but the result code may provide an indication of the reason for the disconnection, and the diagnostic message may provide a human-readable explanation.

A.4.13.7 NOT search filter

A NOT search filter is a type of LDAP search filter that is intended to serve as a container that holds exactly one embedded search filter. The NOT filter is essentially an inverse operation, and in order for an entry to match a NOT filter, it must not match the embedded filter.

NOT filters may be represented as a string by enclosing the entire filter in parentheses and placing an exclamation point just after the opening parentheses. For example, a filter of `(!(objectClass=person))` will only match an entry if it does not have an object class value of `person`.

A.4.13.8 numeric algorithm

A proxy distribution algorithm in which data is split into partitions based on numerical delimitations. For example, [1-1000[for one partition, and [1000-2000[for the next partition.

A.4.13.9 nsuniqueid

A unique identifier that is assigned to each entry in the directory server to resolve naming conflicts while migrating legacy applications using Oracle Directory Server Enterprise Edition as an LDAP database to Oracle Unified Directory.

A.4.14 O

A.4.14.1 object class

An object class is a schema element that correlates an object identifier and a set of names with a set of required and optional attribute type.

The components of an object class definition include:

- An OID used to uniquely identify the object class.
- A set of zero or more names that can be used to more easily reference the object class.
- An optional superior class, which may define additional required and optional attribute types.
- An optional object class type value that indicate whether the object class is structural object class, auxiliary object class, or abstract object class.

- An optional set of one or more attribute type names or OIDs for attributes that must be present in entries containing the object class.
- An optional set of one or more attribute type names or OIDs for attributes that may optionally be present in entries containing the object class.

Every entry must have exactly one structural object class, and it may have zero or more auxiliary classes. The complete set of object classes in an entry define the set of attribute types that are required or allowed to be present. You can also use the structural class to link the entry with one or more of the following:

- name form
- DIT content rule
- DIT structure rule

The set of object classes defined in the server may be determined by retrieving the `objectClasses` attribute of the subschema subentry. For more information about object classes, see Understanding Object Classes.

A.4.14.2 object class type

An object class type is used to define the category for an object class. There are three object class type values:

- structural object class

  A structural object class is used to define the primary type for an entry. Each entry must have exactly one structural class, and it defines the core type of object that the entry represents.

- auxiliary object class

  An auxiliary object class is used to define a characteristic of an entry. An entry may have zero or more auxiliary classes. The set of auxiliary classes that an entry may have may be controlled by a DIT content rule that is associated with the entry's structural class.

- abstract object class

  An abstract object class is not intended to be used directly in entries but should be subclassed by a structural or auxiliary class.

The inheritance model used for LDAP object classes is very similar to the inheritance model for Java classes. Just like an entry must only exactly one structural object class, a Java class must have exactly one superclass. Similarly, while an entry may have multiple auxiliary classes, a Java class may implement multiple interfaces. Finally, it is not possible to instantiate an abstract Java class, just as it is not possible to create an entry containing only an abstract object class.

A.4.14.3 object identifier

An object identifier (OID) is a string that comprises a series of integers separated by periods. It is used as a unique identifier for various types of elements in the Directory Server, including:

- attribute syntax
- matching rule
- attribute type
- object class
- name form

- control
- extended operation
- supported feature

A.4.14.4 operation ID

An operation ID is an integer identifier that is assigned to each operation performed on a client connection. It is used primarily for logging purposes, so that it is possible to correlate a response log message with the corresponding request message.

The first operation performed on a client connection is assigned an operation ID of zero, and it is incremented by one for each additional request received on that client connection.

A.4.14.5 operational attribute

An operational attribute is an attribute type with an attribute usage of `directoryOperation`, `distributedOperation`, or `dSAOperation`. Operational attributes are used for storing information needed for processing by the server itself or for holding any other data maintained by the server that was not explicitly provided by clients.

Operational attributes are not included in entries returned from search operations unless they are explicitly included in the list of search attributes. An explicit value of `+` (the plus sign) may also be included to request that all operational attributes be returned.

A.4.14.6 ordering index

An ordering index is a type of index that is used to keep track of the relative order of values for an attribute. It is very similar to an equality index except that it uses an ordering matching rule instead of an equality matching rule to normalized value the values. Ordering indexes may not be maintained for attributes that do not have a corresponding ordering matching rule.

A.4.14.7 OR search filter

An OR search filter is a type of LDAP search filter that is intended to serve as a container that holds zero or more other search filters. In order for an entry to match an OR filter, it must match at least one of the filters contained in that OR filter.

OR filters may be represented as a string by enclosing the entire filter in parentheses and placing a pipe symbol (|) just after the opening parenthesis. For example, a filter of `(|(uid=john.doe)(uid=jane.doe))` represents an OR search filter that embeds the `(uid=john.doe)` and `(uid=jane.doe)` equality filters.

An OR filter that does not contain any embedded filters is called an LDAP false filter. The string representation for an LDAP false filter is `(|)`, and LDAP false filters will never match any target entry.

A.4.14.8 OID Search Count Request Control

The OID Search Count Request Control does not contain any data. It must be sent with a search request.

A.4.14.9 OID Search Count Response Control

The OID Search Count Response Control data contains a BER-encoded integer that represents the number of entries corresponding to the search. No entry is returned from the search. Only the control is returned indicating the number of entries corresponding to that search.

A.4.15 P

A.4.15.1 partition

In a proxy distribution deployment, the data is split into smaller chunks of data, each of which is known as a partition. A partition of data is typically stored on a separate remote LDAP server, or on a set of replicated remote LDAP servers to ensure high availability.

A.4.15.2 password

A password is a secret value that may be used to provide proof of identity in some authentication mechanisms. In particular, a password is used in simple authentication, as well as the CRAM-MD5 SASL mechanism, DIGEST-MD5 SASL mechanism, and PLAIN SASL mechanism Simple Authentication and Security Layer mechanisms.

The security that a password provides is based entirely on the fact that only the password's owner knows what the password is. If someone else learns a user's password through some means, then that third party can impersonate that user and may be able to perform any operation available to that user.

The Directory Server provides several password policy features that can be used to help ensure that passwords are not discovered by third-party individuals (for example, helping to ensure that users are not allowed to use weak passwords, providing protection against brute-force attacks, requiring authentication attempts and password changes from being performed in a secure manner), but nevertheless passwords are often considered weaker forms of protection than other kinds of identification like certificate.

A.4.15.3 password expiration

Password expiration is an element of the Directory Server password policy that can be used to limit the length of time that a user can continue to use the same password. If password expiration is enabled, once a user changes his or her password, they can use it for a length of time specified as the maximum password age. As the password expiration time draws near, the user may receive warning messages in the form of control in the bind response. Once the password has expired, the user will no longer be allowed to authentication.

Once the user's password has expired, it may be necessary for an administrator to password reset before the account may be used. Alternately, if the password policy is configured appropriately, the user may also be able to change their own expired password using the Password Modify extended operation.

A.4.15.4 password generator

A password generator is a piece of logic that may be used to generate a password for a user as part of a Password Modify extended operation. It will be used if the password modify request does not include a new password.

A.4.15.5 Password Modify extended operation

The Password Modify extended operation is a type of extended operation that may be used to change or password reset user password. It is defined in RFC 3062 (http://www.ietf.org/rfc/rfc3062.txt) and both the request and response operations have an OID of 1.3.6.1.4.1.4203.1.11.1.

The value for the password modify request is:

```
PasswdModifyRequestValue ::= SEQUENCE {
     userIdentity    [0]  OCTET STRING OPTIONAL
     oldPasswd       [1]  OCTET STRING OPTIONAL
     newPasswd       [2]  OCTET STRING OPTIONAL }
```

The value for the password modify response is:

```
PasswdModifyResponseValue ::= SEQUENCE {
     genPasswd       [0]     OCTET STRING OPTIONAL }
```

A.4.15.6 password policy

The Directory Server password policy provides a mechanism for controlling how passwords will be stored and maintained in the server, and how users will be allowed to authenticate.

Elements of the password policy include:

- The attribute used to store user passwords. By default, this is the `userPassword` attribute.
- The default set of password storage scheme that will be used to encode passwords stored in the server.
- A set of deprecated password storage scheme that can be used to authenticate users but cause the password to be re-encoded using the default schemes upon a successful bind.
- A flag that indicates whether users will be allowed to change their own passwords.
- A number of settings related to password expiration, including the maximum age for passwords, warnings before expiration, and whether users will be allowed to change their passwords after they expire.
- A number of settings related to account lockout, which can be used to prevent users from authenticating after too many failed attempts.
- Flags that indicate whether users will be required to change their passwords the first time they authenticate, whether they will be required to change their passwords after they have been reset by an administrator, or both.
- A set of password validator that can be used to determine whether proposed new password values are acceptable for use.
- A flag that indicates whether users will be required to provide their current passwords to be allowed to change their passwords.
- A flag that indicates whether clients will be allowed to specify new passwords that have already been encoded using one of the password storage schemes defined in the server.

Allowing pre-encoded passwords may be necessary for some applications, but may allow the user to bypass certain restrictions, like password validators, that might otherwise be enforced.

- Settings related to maintaining the last login time, including the attribute to use to store its value, the format to use for the time stamp, and whether to lock an account after too much time has elapsed without authenticating.
- Flags that control whether the user will be required to authenticate in a secure manner, whether they will be required to change their passwords in a secure manner, or both.

A.4.15.7 password policy control

The password policy request control is a type of LDAP control that can be used to request information about the current password policy state for a user entry. It is defined in draft-sisbehera-ldap-password-policy (https://tools.ietf.org/html/draft-behera-ldap-password-policy-10). Both the request and response controls have an OID of 1.3.6.1.4.1.42.2.27.8.5.1. The request control does not have a value. The response control value is encoded as follows:

```
PasswordPolicyResponseValue ::= SEQUENCE {
     warning [0] CHOICE {
          timeBeforeExpiration [0] INTEGER (0 .. maxInt),
          graceAuthNsRemaining [1] INTEGER (0 .. maxInt) } OPTIONAL,
     error   [1] ENUMERATED {
          passwordExpired             (0),
          accountLocked               (1),
          changeAfterReset            (2),
          passwordModNotAllowed       (3),
          mustSupplyOldPassword       (4),
          insufficientPasswordQuality (5),
          passwordTooShort            (6),
          passwordTooYoung            (7),
          passwordInHistory           (8) } OPTIONAL }
```

For an example of using this control in a search request, see Searching Using the Password Policy Control.

A.4.15.8 password reset

A password reset is the act of a server administrator changing a user's password. A password reset is a password change that is performed by any user other than the one that owns the account.

A.4.15.9 password storage scheme

A password storage scheme provides a mechanism for encoding user passwords for storage in the server. In most cases, the password is encoded in a manner that prevents users from determining what the clear-text password is, while still allowing the server to determine whether the user-supplied password is correct.

> **Note:**
>
> Old or weak hashing algorithms like MD5, RC4, SHA1 are listed here considering the backward compatibility. Oracle strongly recommends the use of newer or stronger algorithms for password storage scheme.

Password storage schemes currently available for use include:

- 3DES

  The password will be encoded using triple DES. Triple DES is a variation of the Data Encryption Standard (DES) that is three times slower than its predecessor but provides stronger reliability. The algorithm uses three 64-bit keys for a combined key length of 192 bits. The data is encrypted with the first key, decrypted with the second key, and then re-encrypted with the third key. You must ensure that all three keys, the first and the second key, or the second and the third keys are not identical.

- AES

  The Advanced Encryption Standard uses a symmetric block cipher that processes data blocks of 128 bits, using cipher keys with lengths of 128 (AES-128), 192 (AES-192), and 256 (AES-256) bits and is based on the Rijndael algorithm

- BASE64

  The password will be base64 encoding, which provides a very weak form of protection and should only be used for cases in which clients require this storage scheme.

- BlowFish

  The password will be encoded using the BlowFish Algorithm with a 128 bits key length.

- CLEAR

  The password will be stored in clear-text without any form of obfuscation. This scheme contains only an implementation for the user password syntax, with a storage scheme name of `{CLEAR}`. Therefore, it does not provide any protection at all, and so this scheme should only be used for cases in which clients require this storage scheme.

  However, you can configure the `ClearPassowrdScheme` configuration parameter to make the server obfuscate the scheme name in curly brackets when it returns the password. This configuration parameter specifies whether the Clear Password Storage scheme obfuscates the scheme name or not.

  You can configure the obfuscate flag to `true`, if you want the server to obfuscate the scheme name. The default value is `false`.

- CRYPT

  The password will be encoded using the crypt algorithm. The crypt algorithm is a one-way algorithm that supports encoding user passwords on Linux and UNIX systems.

  The newer modular crypt algorithms, which support MD5, SHA256, and SHA512, are more secure than the UNIX crypt algorithm. The UNIX crypt algorithm is considered weak by current standards and should generally be used only for users who require this password storage scheme. However, to ensure compatibility with existing deployments, the UNIX crypt algorithm is the default algorithm for the CRYPT password storage scheme.

- MD5

  The password will be encoded using an unsalted version of the MD5 message digest algorithm. This is relatively secure, although a salt hash is preferred, and one of the Secure Hash Algorithm variants are considered stronger than MD5.

- PBKDF2 HMAC SHA-1

  The password will be encoded using Password Based Key Derivation Function 2 (PBKDF2) with keyed-hash message authentication code (HMAC) SHA-1.

- PBKDF2 HMAC SHA-256

The password will be encoded using Password Based Key Derivation Function 2 (PBKDF2) with keyed-hash message authentication code (HMAC) SHA-256.

- PBKDF2 HMAC SHA-512

  The password will be encoded using Password Based Key Derivation Function 2 (PBKDF2) with keyed-hash message authentication code (HMAC) SHA-512.

> **Note:**
>
> Oracle Unified Directory Server uses default values for the iteration count and the number of bytes for salt for PBKF2 HMACSHA based password storage schemes. If needed, the iteration count and the number of bytes for salt used for the PBKF2 HMACSHA based password storage scheme can be updated by using the `dsconfig set-password-storage-scheme-prop --advanced` command (property names are: `pbkdf2hmacsha-iteration-count`, `pbkdf2hmacsha-num-salt-bytes`).
>
> This configuration step must be performed with caution. If any of the PBKDF2 based schemes is used in the Oracle Unified Directory instance and there are existing user passwords making use of any of these schemes, then validations of those older passwords would fail after the configuration update and those older passwords would need to be reset. Oracle Unified Directory Server needs a restart when the iteration count or the number of bytes for salt is updated.

- RC4

  The password will be encoded using RC4, a stream cipher using a variable key-size stream cipher with byte-oriented operations. The algorithm is based on the use of a random permutation.

- Salted MD5

  The password will be encoded using a salted version of the MD5 message digest algorithm.

- Salted SHA-1

  The password will be encoded using a salted version of the SHA-1 Secure Hash Algorithm. This is the default password storage scheme used by the directory server.

- Salted SHA-256

  The password will be encoded using a salted 256-bit version of the SHA-2 Secure Hash Algorithm.

- Salted SHA-384

  The password will be encoded using a salted 384-bit version of the SHA-2 Secure Hash Algorithm.

- Salted SHA-512

  The password will be encoded using a salted 512-bit version of the SHA-2 Secure Hash Algorithm.

- SHA

  The password will be encoded using an unsalted version of the SHA-1 Secure Hash Algorithm. The salted variant of this algorithm is preferred.

- SHA-256

The password will be encoded using an unsalted version of the SHA-256 Secure Hash Algorithm. The salted variant of this algorithm is preferred.

- SHA-512

  The password will be encoded using an unsalted version of the SHA-512 Secure Hash Algorithm. The salted variant of this algorithm is preferred.

- User-defined
  User-defined password storage scheme in Oracle Unified Directory (OUD) provides the ability to implement and deploy custom password hashing schemes into the server. By implementing a custom scheme, you can control the way passwords are stored and managed in OUD. This framework provides an ability to implement schemes which are not available out of the box in OUD. To use user-defined password storage scheme, see the *Building and Deploying User-defined Password Storage Scheme in Oracle Unified Directory* section in the Developing Plug-Ins for Oracle Unified Directory guide.

- EUS PBKDF2 SHA-512

  This password storage scheme will provide a mechanism for encoding user passwords using SHA-512 based proprietary algorithm designed for Oracle RDBMS. The password will be encoded using the same algorithm that is used to generate Oracle Database 12c (`Multi-Round SHA-512`) password verifier.

Be aware that the directory server also supports the use of the authentication password syntax.

A.4.15.10 password validator

A password validator is a component of the directory server password policy that is used to determine whether a proposed password is acceptable for use. The directory server provides an extensible API for developing custom password validators, but it does come with several different types of password validators, including:

- A validator that can be used to reject a password if the value exists in any of the attribute contained in the user's entry.
- A validator that can be used to reject a password if the value does not contain characters from an acceptable range of character sets.
- A validator that can be used to reject a password if it is a word that can be found in a dictionary.
- A validator that can be used to reject a password if it is too long or too short.
- A validator that can be used to reject a password if it contains a string of too many repeated characters.
- A validator that can be used to reject a password if it is too similar to the user's current password.
- A validator that can be used to reject a password if it does not contain enough unique characters.

A.4.15.11 persistent search control

The persistent search control is a type of LDAP control that may be used for clients to be notified of changes to entry that match the criteria from the associated LDAP search operation. The persistent search control is described in `draft-ietf-ldapext-psearch` and has an OID of 2.16.840.1.113730.3.4.3. It is defined as follows:

```
PersistentSearch ::= SEQUENCE {
     changeTypes INTEGER,
     changesOnly BOOLEAN,
     returnECs BOOLEAN
}
```

search result entry returned as part of this search may optionally include the entry change notification control to describe the way in which the entry changed. For an example of using this control in a search, see Searching Using the Persistent Search Control.

A.4.15.12 PLAIN SASL mechanism

The PLAIN Simple Authentication and Security Layer mechanism provides a way for clients to authentication to the Directory Server with a username and password. In general, it is very similar to simple authentication, except that the client can identify itself with a username rather than a distinguished name. It also provides the ability for the client to specify an alternate authorization ID.

Like simple authentication, the PLAIN SASL mechanism does not provide any form of protection for the user password, so it may be advisable to only use this authentication method over secure communication channels like those provided by Secure Sockets Layer or StartTLS extended operation.

A.4.15.13 plug-in

A plug-in is a piece of code that can be used to interject some custom logic into the way that the Directory Server performs its processing. The directory server supports several different types of plug-ins, including:

- Pre-parse plug-ins, which allow the server to alter the contents of a request before the server begins processing on it. Pre-parse plug-ins are available for all types of operations.
- Pre-operation plug-ins, which allow the server to take some action just before the core processing for an operation. Pre-operation plug-ins are available for all types of operations except abandon operation and unbind operation.
- Post-operation plug-ins, which allow the server to take some action just after the core processing for an operation but before the response has been sent to the client (it may be used to alter the response to the client). Post-operation plug-ins are available for all types of operations.
- Post-response plug-ins, which allow the server to take some action after all other processing for an operation has completed. Post-response plug-ins are available for all types of operations except abandon and unbind.
- Search result entry plug-ins, which alter the contents of a search result entry being sent as part of a search operation.
- Search result reference plug-ins, which alter the contents of a search result reference being sent as part of a search operation.
- Intermediate response plug-ins, which alter the contents of an LDAP intermediate response being sent to a client.
- Startup plug-ins, which perform some processing when the server is first starting.
- Shutdown plug-ins, which perform some processing when the server is performing a graceful shutdown.
- Post-connect plug-ins, which perform some processing as part of accepting a new client connection.

- Post-disconnect plug-ins, which perform some processing immediately after a connection is terminated.
- LDIF import plug-ins, which alter the contents of entry being imported from an LDAP Data Interchange Format file.
- LDIF export plug-ins, which alter the contents of entries being exported from a server back end.

A.4.15.14 presence index

A presence index is a type of index that is used to keep track of the entries that have at least one value for a specified attribute. There is only a single presence index key per attribute, and its ID list contains the entry ID for all entries that contain the specified attribute.

A.4.15.15 presence search filter

A presence search filter is a type of LDAP search filter that can be used to identify entries that have at least one value for a specified attribute. The string representation of an LDAP presence filter comprises an opening parenthesis followed by the attribute name, an equal sign, an asterisk, and the closing parenthesis. For example, an equality filter of `(aci=*)` will match any entry containing at least one value for the `aci` attribute.

A.4.15.16 privilege

The directory server provides a privilege subsystem, which can be used to define capabilities that will be granted to users. The privilege subsystem works with the access control implementation in the process of determining whether a user will be allowed to perform a certain operation.

Some privileges defined in the directory server include:

- bypass-acl

  Allows the user to bypass access control evaluation
- modify-acl

  Allows the user to modify access control rule defined in the server.
- config-read

  Allows the user to have read access to the server configuration
- config-write

  Allows the user to have write access to the server configuration
- server-shutdown

  Allows the user to request that the server shut down
- server-restart

  Allows the user to request that the server perform an in-core restart
- proxied-auth

  Allows the user to request an operation with a different authorization ID
- unindexed-search

  Allows the user to request an unindexed search
- password-reset

Allows the user to password reset for other users

- update-schema

  Allows the user to update the server schema

See Root Users and the Privilege Subsystem for more information on the privilege subsystem.

A.4.15.17 proportional algorithm

A proxy load balancing algorithm in which client requests are distributed to a set of replicated remote LDAP servers. How many requests are sent to each remote LDAP server is determined by the weight set.

A.4.15.18 protocol data unit

A protocol data unit (PDU) is a single complete element of network communication. For LDAP, the PDU is the message.

A.4.15.19 protocol op

The protocol op is the element in the message that contains the heart of the request or response. That is, it indicates what type of message it is. There are several different kinds of protocol op elements, including:

- The abandon operation
- The add operation
- The bind operation
- The compare operation
- The delete operation
- The extended operation
- The modify operation
- The modify DN operation
- The search operation
- The unbind operation
- The LDAP intermediate response

A.4.15.20 proxied authorization control

The proxied authorization control is a type of control that can be used to request that the associated operation be performed under the authorization of another user.

There are actually two different forms of the proxied authorization control, both of which are request controls that may be attached to an add operation, compare operation, delete operation, modify operation, modify DN operation, or search operation operation.

The proxied authorization v1 control is defined in early versions of draft-weltman-ldapv3-proxy. It has an OID of 2.16.840.1.113730.3.4.12 and the control value should be encoded as:

```
proxyAuthValue::= SEQUENCE {
      proxyDN LDAPDN
}
```

The proxied authorization v2 control is defined in RFC 4370 (http://www.ietf.org/rfc/rfc4370.txt). It has an OID of 2.16.840.1.113730.3.4.18 and the value is a string containing the desired authorization ID.

For an example of using this control in a search request, see Searching Using the Proxied Authorization Control..

A.4.16 Q

A.4.16.1 quality of protection

Quality of protection (QoP) is a property of certain Simple Authentication and Security Layer mechanisms (especially the DIGEST-MD5 SASL mechanism and GSSAPI SASL mechanism mechanisms) that can be used to protect the communication between the client and the server.

There are three different QoP levels:

- auth

  This indicates that the associated SASL mechanism should only be used to authenticate the client connection. It should not provide any other protection for the client-server communication

- auth-int

  This indicates that the associated SASL mechanism should be used for authentication, and then should also provide integrity protection for the communication between the client and server. Integrity protection will not prevent third-party observers from understanding the communication, but it will ensure that a man-in-the-middle cannot alter that communication in an undetectable manner

- auth-conf

  This indicates that the associated SASL mechanism should be used for authentication, and then should also provide integrity and confidentiality protection for the communication between the client and the server. This will ensure that third-party observers will be unable to understand the communication

Currently, the directory server supports only the `auth` quality of protection. It does not support either the `auth-int` or `auth-conf` levels.

A.4.17 R

A.4.17.1 real attributes only control

The real attributes only control is a control that may be used to request that the server only include real attributes in matching entries. That is, virtual attribute are excluded from search result entry.

The real attributes only control has a request object identifier of `2.16.840.1.113730.3.4.17` and no value.

In the following search, the `numsubordinates` virtual attribute is requested and returned:

```
$ ldapsearch -D "cn=directory manager" -j pwd-file -b "ou=people,dc=example,dc=com" \
  -s base "objectclass=*" numsubordinates
```

```
version: 1
dn: ou=People,dc=example,dc=com
numSubordinates: 50
```

In the following search, the `numsubordinates` virtual attribute is requested but is not returned because the real attributes only control is used:

```
$ ldapsearch -D "cn=directory manager" -j pwd-file -J "2.16.840.1.113730.3.4.17" \
  -b "ou=people,dc=example,dc=com" -s base "objectclass=*" numsubordinates
version: 1
dn: ou=People,dc=example,dc=com
```

A.4.17.2 referential integrity

Referential integrity is a mechanism for ensuring that any references to an entry are updated whenever that entry is removed or altered. Historically, referential integrity is primarily used to ensure that attributes with a distinguished name syntax (especially group membership attributes like `member` and `uniqueMember`) are properly maintained for delete operation and modify DN operation operations. For a delete operation, any references to the target entry will be removed. For modify DN operations, any references to the target entry will be renamed accordingly.

The directory server provides a configurable referential integrity plug-in that you can install using the dsconfig command.

A.4.17.3 referral

A referral provides a reference to an alternate location in which an operation may be processed. A referral may be included in an LDAP result object with a result code of `10` and an appropriate set of LDAP URL. It may also be returned to clients in a search result reference.

A.4.17.4 relative distinguished name

A relative distinguished name, or RDN, is a single component within a distinguished name. It comprises one or more name-value pairs, in which the name and the value are separated by an equal sign (for example, for an RDN of `uid=ann`, the name is `uid` and the value is `ann`), and if there are multiple name-value pairs then they should be separated by plus signs (for example, for an RDN of `cn=John Doe+employeeNumber=12345`, the name-value pairs are `cn=John Doe` and `employeeNumber=12345`). In practice, RDNs containing multiple name-value pairs (called "multivalued RDNs") are rare, but they can be useful at times when either there is no unique attribute in the entry or you want to ensure that the entry's DN contains some useful identifying information.

Even though a DN may be composed of multiple RDN components, the leftmost component is typically referred to as the entry's RDN. For example, in a DN of `uid=john.doe,ou=People,dc=example,dc=com`, the RDN would be `uid=john.doe`. The attribute values specified in an entry's RDN must be contained in that entry, so the entry `uid=john.doe,ou=People,dc=example,dc=com` must have a `uid` value of `john.doe`.

A.4.17.5 replica

A replica is a Directory Server instance that participates in replication.

A.4.17.6 replication

Replication is a form of data synchronization that is used to ensure that changes in the directory environment are reflected in each instance of the server. That is, whenever a change is made in one Directory Server instance, that same change is also made in every other instance. Replication typically occurs where the source and destination are in the same product; for example, both are Oracle Unified Directory.

A.4.17.7 replication repair control

The replication repair control is a control that can be used to resolve replication inconsistencies on a single server in a topology.

The replication repair control has a request object identifier of `1.3.6.1.4.1.26027.1.5.2` and no value.

For an example of using the replication repair control, see Detecting and Resolving Replication Inconsistencies.

A.4.17.8 request for comments

A request for comments (RFC) is an IETF (`http://www.ietf.org/`) specification that has been promoted from an Internet Draft and may be considered significantly more stable than drafts.

A.4.17.9 restore

A restore operation provides a mechanism for replacing the contents of a Directory Server back end with information taken from a previous backup. It can serve as a disaster recovery mechanism, and in some cases can be used for binary copy initialization of a replica.

A.4.17.10 result

See LDAP result.

A.4.17.11 result code

A result code is an integer value that provides general information about the result of the operation. Defined result codes include:

Value	Name	Description
0	Success	Indicates that the associated operation completed successfully.
1	Operations Error	Indicates that the associated request was out of sequence with another operation in progress (for example, a non-bind request in the middle of a multi-stage SASL bind).
2	Protocol Error	Indicates that the client sent data to the server that did not comprise a valid LDAP request.
3	Time Limit Exceeded	Indicates that processing on the associated request was terminated because it took too long to complete. For a search operation, perhaps some matching entries had been returned when the time limit was reached.

Value	Name	Description
4	Size Limit Exceeded	Indicates that there were more entries matching the criteria contained in a search operation than were allowed to be returned by the size limit configuration.
5	Compare False	Indicates that a compare operation completed successfully, but the provided attribute value assertion did not match the target entry.
6	Compare True	Indicates that a compare operation completed successfully, and the provided attribute value assertion matched the target entry.
7	Auth Method Not Supported	Indicates that the Directory Server does not support the requested authentication method.
8	Strong Auth Required	Indicates that the Directory Server requires that the client use a strong authentication mechanism.
10	Referral	Indicates that the requested operation could not be processed in the target server but may be attempted in elsewhere.
11	Admin Limit Exceeded	Indicates that processing on the requested operation could not be completed because an administrative limit was reached. For a search operation, it is possible that some matching entries had been returned when the administrative limit was reached.
12	Unavailable Critical Extension	Indicates that the request included a critical control that could not be processed by the server.
13	Confidentiality Required	Indicates that the requested operation requires a secure communication channel between the client and the server.
14	SASL Bind In Progress	Indicates that a SASL bind operation requires multiple stages and the response containing this result code is one of the intermediate stages.
16	No Such Attribute	Indicates that the associated request targeted an attribute or attribute value that does not exist in the specified entry.
17	Undefined Attribute Type	Indicates that the associated request included an attribute type that is not defined in the server schema.
18	Inappropriate Matching	Indicates that the associated search request included a filter with a component targeting an attribute type for which no appropriate matching rule is defined.
19	Constraint Violation	Indicates that the requested operation could not be completed because it would have violated some constraint defined in the server (for example, it would have duplicated a value for a unique attribute).
20	Attribute or Value Exists	Indicates that an operation attempted to create an attribute value in an entry that already existed in the entry, or that it attempted to create an additional value for a single-valued attribute.
21	Invalid Attribute Syntax	Indicates that requested operation attempted to specify a value that violated the syntax for the associated attribute type.
32	No Such Object	Indicates that the requested operation targeted an entry that does not exist in the server.
33	Alias Problem	Indicates that an operation targeted an alias entry and that operation is not allowed on alias entries.

Value	Name	Description
34	Invalid DN Syntax	Indicates that the requested operation included an entry DN that was malformed.
35	Is Leaf	Indicates that the requested operation targeted a leaf entry but the operation requires a non-leaf entry.
36	Alias Dereferencing Problem	Indicates that the associated search operation encountered an alias that could not be properly dereferenced.
48	Inappropriate Authentication	Indicates that the client attempted to bind in a manner that is inappropriate for the target user (for example, the user attempted simple authentication but does not have a password).
49	Invalid Credentials	Indicates that the client attempted to authenticate with invalid credentials (for example, the target DN or password was incorrect).
50	Insufficient Access Rights	Indicates that the client was not allowed to perform the requested operation.
51	Busy	Indicates that the server is too busy to process the requested operation.
52	Unavailable	Indicates that the server is unavailable for processing operations.
53	Unwilling to Perform	Indicates that the server is unwilling to perform the requested operation for some reason.
54	Loop Detect	Indicates that the server encountered a loop of some type (for example, a chaining loop or an alias loop).
60	Sort Control Missing	Indicates that the client requested a search operation containing the virtual list view control that did not also include the server-side sort control.
61	Offset Range Error	Indicates that the request included a virtual list view control that specified an invalid offset (for example, one that was beyond the end of the result set).
64	Naming Violation	Indicates that the operation attempted to create an entry with a DN that violated a naming constraint (for example, using an RDN attribute that is not allowed by the associated name form).
65	Object Class Violation	Indicates that the operation attempted to create or modify an entry so that the set of attributes it contained were in violation of the associated object class definitions (for example, it included an attribute that was not allowed or was missing a required attribute).
66	Not Allowed On Nonleaf	Indicates that the associated operation was not allowed on non-leaf entries (for example, an attempt to delete an entry that has one or more subordinate entries).
67	Not Allowed On RDN	Indicates that the associated operation is not allowed on the RDN attribute for an entry.
68	Entry Already Exists	Indicates that the add or modify DN operation would have resulted in an entry with a DN that already exists in the server.
69	Object Class Mods Prohibited	Indicates that the requested operation attempted to alter the structural object class for the entry in a manner that was not allowed.

Value	Name	Description
71	Affects Multiple DSAs	Indicates that the requested operation would have impacted multiple servers (for example, a modify DN operation would have moved an entry from one server to another through a chained back end).
76	Virtual List View Error	Indicates that the associated search operation could not be completed successfully because a problem occurred while processing the virtual list view request.
80	Other	This indicates that the operation failed for some reason that is not more appropriately classified by any other defined result code.
81	Server Down	This is a client-side result code that is used to indicate that the client detected that an established connection was no longer available.
82	Local Error	This is a client-side result code that is used to indicate that some client-side problem occurred that prevented it from completing the associated processing successfully.
83	Encoding Error	This is a client-side result code that is used to indicate that an error occurred while attempting to encode the request to send to the server.
84	Decoding Error	This is a client-side result code that is used to indicate that an error occurred while attempting to decode the response received from the server.
85	Timeout	This is a client-side result code that is used to indicate that the client did not receive a response in an acceptable length of time.
86	Authentication Type Unknown	This is a client-side result code that is used to indicate that the client does not support the requested authentication method.
87	Filter Error	This is a client-side result code that is used to indicate that a provided filter string could not be parsed as a valid filter.
88	User Canceled	This is a client-side result code that is used to indicate that the client canceled the request.
89	Parameter Error	This is a client-side result code that is used to indicate that there was a problem with a parameter provided for a request element.
90	No Memory	This is a client-side result code that is used to indicate that the client ran out of memory while attempting to process the requested operation (for example, while queueing the search result entries).
91	Connect Error	This is a client-side result code that is used to indicate that the client could not establish a connection to the target server.
92	Not Supported	This is a client-side result code that is used to indicate that the requested operation is not supported by the client.
93	Control Not Found	This is a client-side result code that is used to indicate that a response did not include an expected control.
94	No Results Returned	This is a client-side result code that is used to indicate that the server did not return any results for a search request when at least one was expected.

Value	Name	Description
95	More Results to Return	This is a client-side result code that is used to indicate that there are more results to return than those that have already been retrieved.
96	Client Loop	This is a client-side result code that is used to indicate that the client detected a referral loop.
97	Referral Limit Exceeded	This is a client-side result code that is used to indicate that the client received too many referrals in the course of processing a request.
100	Invalid Response	This is a client-side result code that is used to indicate that the result received for the associated operation is invalid.
101	Ambiguous Response	This is a client-side result code that is used to indicate that the result received from the server was ambiguous (for example, there was more than one response received fro the associated operation).
112	TLS Not Supported	Indicates that the server does not support the StartTLS extended operation.
113	Intermediate Response	Indicates intermediate response messages sent by the server in the course of processing the request.
114	Unknown Type	Indicates that the server received a request with an invalid or unknown protocol op type.
118	Canceled	Indicates that the server canceled processing on the request at the request of the client.
119	No Such Operation	Indicates that the client attempted to cancel a request that was unknown to the server (for example, because it had already completed processing).
120	Too Late	Indicates that the client attempted to cancel a request that had already been processed beyond a point at which it could no longer be canceled.
121	Cannot Cancel	Indicates that the client attempted to cancel an operation that could not be canceled (for example, a bind, unbind, abandon, cancel, or StartTLS request).
122	Assertion Failed	Indicates that the associated operation was not processed because the request included an LDAP assertion control with an assertion filter that did not match the target entry.
123	Authorization Denied	Indicates that the associated operation was not processed because the request included a proxied authorization control but the client was not allowed to use that control.

A.4.17.12 root DN

A root DN (or root user) is a type of account that exists in the Directory Server which is generally given full access to all data in the server, much like the root user in UNIX systems. Root users by default will be allowed to bypass access control evaluation, will have full access to the server configuration, and perform most other types of operations.

The directory server is different from most other servers regarding root users in two key ways:

- You can configure the directory server with multiple root users, which enables each root user to have a different set of credentials. It also enables each administrator to have a separate, independent root account rather than a single account that is shared by all administrators.

- All of the rights given to root users are assigned through privilege. Using the privilege subsystem, it is possible to create non-root users with some or all of the capabilities normally available only to root users. It is also possible to take away privileges from root users if so desired.

For more information on root users and the privilege subsystem, see Root Users and the Privilege Subsystem.

A.4.17.13 root DSE

The root DSE is a special entry that provides information about the contents and capabilities of the server. The distinguished name is a zero-length string with no relative distinguished name components, also called the null DN.

The attribute contained in the root DSE include:

- namingContexts

  Lists the naming context for the server

- supportedAuthPasswordSchemes

  Lists the object identifier of the supported password storage scheme using the authentication password syntax

- supportedControl

  Lists the OIDs of the supported control in the server

- supportedExtension

  Lists the OIDs of the supported extension in the server

- supportedFeatures

  Lists the OIDs of the supported feature in the server

- supportedSASLMechanisms

  Lists the OIDs of the supported Simple Authentication and Security Layer mechanisms in the server

- vendorName

  Provides the name of the vendor for the server

- vendorVersion

  Provides a product version string

The following example demonstrates how to use the `ldapsearch` command to read the root DSE. In this example the file `/tmp/pwd.txt` contains the Directory Manager password. The server is listening for LDAP requests on port 1389.

```
$ ldapsearch -D "cn=Directory Manager" -j /tmp/pwd.txt -p 1389 -b "" \
  -s base "(objectclass=*)" +
dn:
supportedLDAPVersion: 2
supportedLDAPVersion: 3
supportedExtension: 1.3.6.1.4.1.4203.1.11.3
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.26027.1.6.1
supportedExtension: 1.3.6.1.4.1.26027.1.6.3
supportedExtension: 1.3.6.1.4.1.26027.1.6.2
supportedExtension: 1.3.6.1.1.8
supportedExtension: 1.3.6.1.4.1.1466.20037
```

```
vendorName: Oracle Corporation
entryDN:
ds-private-naming-contexts: cn=admin data
ds-private-naming-contexts: cn=ads-truststore
ds-private-naming-contexts: cn=backups
ds-private-naming-contexts: cn=config
ds-private-naming-contexts: cn=monitor
ds-private-naming-contexts: cn=schema
ds-private-naming-contexts: cn=tasks
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.840.113556.1.4.473
supportedControl: 1.2.840.113556.1.4.805
supportedControl: 1.3.6.1.1.12
supportedControl: 1.3.6.1.1.13.1
supportedControl: 1.3.6.1.1.13.2
supportedControl: 1.3.6.1.4.1.26027.1.5.2
supportedControl: 1.3.6.1.4.1.42.2.27.8.5.1
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.2
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.8
supportedControl: 1.3.6.1.4.1.4203.1.10.2
supportedControl: 1.3.6.1.4.1.7628.5.101.1
supportedControl: 2.16.840.1.113730.3.4.12
supportedControl: 2.16.840.1.113730.3.4.16
supportedControl: 2.16.840.1.113730.3.4.17
supportedControl: 2.16.840.1.113730.3.4.18
supportedControl: 2.16.840.1.113730.3.4.19
supportedControl: 2.16.840.1.113730.3.4.2
supportedControl: 2.16.840.1.113730.3.4.3
supportedControl: 2.16.840.1.113730.3.4.9
supportedSASLMechanisms: PLAIN
supportedSASLMechanisms: EXTERNAL
supportedSASLMechanisms: CRAM-MD5
supportedSASLMechanisms: DIGEST-MD5
supportedFeatures: 1.3.6.1.1.14
supportedFeatures: 1.3.6.1.4.1.4203.1.5.1
supportedFeatures: 1.3.6.1.4.1.4203.1.5.2
supportedFeatures: 1.3.6.1.4.1.4203.1.5.3
subschemaSubentry: cn=schema
hasSubordinates: true
entryUUID: d41d8cd9-8f00-3204-a980-0998ecf8427e
numSubordinates: 1
namingContexts: dc=example,dc=com
vendorVersion: Oracle Unified Directory 11.1.1.5.0
supportedAuthPasswordSchemes: MD5
supportedAuthPasswordSchemes: SHA1
supportedAuthPasswordSchemes: SHA256
supportedAuthPasswordSchemes: SHA384
supportedAuthPasswordSchemes: SHA512
```

For more information on how to search the root DSE entry, see Using Advanced Search Features.

A.4.17.14 route

In proxy mode, the path on which requests are sent to the remote LDAP server when using a load balancing algorithm.

A.4.18 S

A.4.18.1 salt

A salt is a collection of random data that may be combined with clear-text data (often a password) that can be used to change the way that it is encoded. In particular, the salt is used to introduce randomness into the encoding process to help thwart dictionary attacks. In general, the salt is appended to the clear-text password, which is the encoded using the desired message digest algorithm, and then the clear-text salt is appended to the message digest and the resulting value is base64 encoding. This makes it possible to determine what the salt was so that it can be used to determine whether a user-supplied password is correct.

The UNIX crypt algorithm uses a relatively weak 12-bit salt, which means that there are only 4096 ways of encoding any value. This is a relatively low number, and therefore it is possible to construct dictionaries of every possible encoding for a wide range of values for use in breaking user passwords. Other password storage scheme in the directory server use a 64-bit salt which provide 18446744073709551616 different ways of encoding any one value.

See also crypt algorithm.

A.4.18.2 saturation algorithm

A proxy load balancing algorithm in which client requests are routed to a priority remote LDAP server. When the main remote LDAP server reaches its *saturation threshold*, the requests are routed to a secondary remote LDAP server.

A.4.18.3 saturation alert

The limit at which a notification is sent to the administrator to indicate that the remote LDAP server is overloaded. Usually, the saturation alert is set higher than the *saturation threshold*.

A.4.18.4 saturation threshold

The saturation threshold is the limit at which the data source is considered overloaded and can no longer handle incoming requests in an optimal way. The saturation threshold is used as part of the proxy saturation algorithm.

A.4.18.5 schema

The schema of a Directory Server defines a set of rules that govern the kinds of information that the server can hold. Directory schema includes many different elements, including:

- attribute syntax

  Provide information about the kind of information that can be stored in an attribute.

- matching rule

  Provide information about how to make comparisons against attribute values.

- matching rule use

  Indicate which attribute types may be used with a particular matching rule.

- attribute type

Define an object identifier and a set of names that may be used to refer to a given attribute, and associates that attribute with a syntax and set of matching rules.

- object class

  Define named collections of attributes and classify them into sets of required and optional attributes.

- name form

  Define rules for the set of attributes that should be included in the relative distinguished name for an entry.

- DIT content rule

  Define additional constraints about the object classes and attributes that may be used with an entry.

- DIT structure rule

  Define rules that govern the kinds of subordinate entries that a given entry may have.

attribute are the elements responsible for storing information in a directory, and the schema defines the rules for which attributes may be used in an entry, the kinds of values that those attributes may have, and how clients may interact with those values.

Clients may learn about the schema elements that the server supports by retrieving an appropriate subschema subentry.

A.4.18.6 schema checking

Schema checking is the process of ensuring that an entry conforms to the constraints defined by the server schema. This includes:

- Make sure the entry contains exactly one structural object class.
- If there is a name form for the entry's structural class, ensure that the relative distinguished name attributes conform with that name form.
- If there is a DIT content rule for the entry's structural class, then ensure that every auxiliary object class is defined.
- Ensure that each object class contained in the entry is defined in the schema.
- Ensure that each attribute contained in the entry is defined in the schema and allowed by the object classes, DIT content rule, or both.
- Ensure that all attributes required by the entry's object classes or DIT content rule are present.
- Ensure that all single-valued attributes contained in the entry only have one value.
- Ensure that the entry's position in the directory information tree (DIT) conforms with DIT structure rule definitions.

A.4.18.7 search attributes

The search attributes element of a search operation provides a way of representing the attribute that should be included in search result entry. In general, the set of search attributes is a list of zero or more attribute description for the attributes to return. If values are specified, then all user attribute and no operational attribute will be returned.

In addition to specific attribute descriptions, the following special values can be provided with various meanings:

- The string `1.1` indicates that no attributes should be included in matching entries.
- The string `*` (the asterisk) indicates that all user attributes should be included in matching entries. This is needed if the server returns all user attributes in addition to one or more operational attributes.
- The string `+` (the plus sign) indicates that all operational attributes should be included in matching entries.
- An object class name can be provided, prefixed with the `@` character. This indicates that all attributes referenced by that object class should be included in matching entries.

A.4.18.8 search base DN

The search base DN is an element of the search operation that works with the search scope to define the subtree of entries that should be considered when processing the search operation. Only entries at or below the search base DN and within the scope will be considered candidates for matching against the LDAP search filter.

A.4.18.9 search filter

See LDAP search filter.

A.4.18.10 search operation

The LDAP search operation can be used to identify entries in the Directory Server that match a given set of criteria. It may return zero or more entries, and also zero or more referrals.

The search request protocol op is defined as follows:

```
SearchRequest ::= [APPLICATION 3] SEQUENCE {
      baseObject      LDAPDN,
      scope           ENUMERATED {
          baseObject              (0),
          singleLevel             (1),
          wholeSubtree            (2),
          ...  },
     derefAliases    ENUMERATED {
          neverDerefAliases       (0),
          derefInSearching        (1),
          derefFindingBaseObj     (2),
          derefAlways             (3) },
     sizeLimit       INTEGER (0 ..  maxInt),
     timeLimit       INTEGER (0 ..  maxInt),
     typesOnly       BOOLEAN,
     filter          Filter,
     attributes      AttributeSelection }
```

The elements of the search request include:

- The search base DN, which specifies the location in the directory information tree (DIT) in which to perform the search.
- The search scope, which specifies the scope of entries at or below the base DN to consider when processing the search.
- The dereference policy to use if any aliases are encountered during processing.
- The size limit, which specifies the maximum number of entries that should be returned from the search (or zero if there should not be any maximum number of entries).

- The time limit, which specifies the maximum length of time in seconds that the server should spend processing the search (or zero if there should not be a maximum number of entries).
- The typesOnly flag, which indicates whether the entries returned should include attribute types only or both types and values.
- The LDAP search filter, which specifies the criteria to use to identify matching entries.
- The search attributes that indicate which attributes should be included in matching entries, or an empty list to indicate that all user attribute should be returned.

There are three types of result elements that can be returned in response to a search request: zero or more search result entry, zero or more search result reference, and exactly one search result done message. The entries and references can be returned in any order (and with search entries and references interspersed), and the search result done message will come last to indicate that there are no more results.

The search result entry protocol op is defined as follows:

```
SearchResultEntry ::= [APPLICATION 4] SEQUENCE {
     objectName      LDAPDN,
     attributes      PartialAttributeList }

PartialAttributeList ::= SEQUENCE OF
                     partialAttribute PartialAttribute
```

Each search result entry includes the DN of the entry and zero or more attributes (potentially including only the attribute type names without the values if the `typesOnly` element of the request is true) as defined in the search attribute list.

The search result reference protocol op is defined as follows:

```
SearchResultReference ::= [APPLICATION 19] SEQUENCE
                          SIZE (1..MAX) OF uri URI
```

Each search result reference includes one or more LDAP URL specifying an alternate location in which the client may search for additional matching entries.

The search result done message is an LDAP result defined as follows:

```
SearchResultDone ::= [APPLICATION 5] LDAPResult
```

A.4.18.11 search result done

A search result done message is a message provided as part of a search operation to indicate that the search has completed and that there will be no more search result entry or search result reference messages.

A.4.18.12 search result entry

A search result entry is an entry returned as part of a search operation. It will contain at least the distinguished name of the entry, and can contain zero or more attributes. The attributes can contain only attribute type names or both types and values (based on the value of the typesOnly flag from the search request). The attributes returned can be based on the search attributes from the client request, but can be pared down based on the server's access control configuration.

A.4.18.13 search result reference

A search result reference provides a mechanism for returning information to clients as part of a search operation that indicates an alternate location in which the client may perform the search to locate additional matching entries. The alternate locations will be specified in the form of LDAP URL.

A.4.18.14 search scope

The LDAP search scope indicates the set of entries at or below the search base DN that may be considered potential matches for a search operation.

There are four defined search scope values:

- baseObject

  This specifies that the search operation should only be performed against the entry specified as the search base DN. No entries below it will be considered.

  Consider a scenario of DIT, which has a baseObject scope with a search base DN of `dc=example,dc=com`.

- singleLevel

  This specifies that the search operation should only be performed against entries that are immediate subordinates of the entry specified as the search base DN. The base entry itself is not included, nor are any entries below the immediate subordinates of the search base entry.

- wholeSubtree

  This specifies that the search operation should be performed against the entry specified as the search base and all of its subordinates to any depth.

- subordinateSubtree

  This specifies that the search operation should be performed against all subordinate entries below the search base to any depth, but the search base entry itself should not be included.

A.4.18.15 Secure Hash Algorithm

The Secure Hash Algorithm (SHA) is a one-way message digest algorithm. There are actually two different forms of the Secure Hash Algorithm:

- SHA-1 is defined in RFC 3174 (`http://www.ietf.org/rfc/rfc3174.txt`) and generates a 160-bit digest.
- SHA-2 is defined in RFC 4634 (`http://www.ietf.org/rfc/rfc4634.txt`) and can be used to generate 256-bit, 384-bit, or 512-bit digests.

All forms of the Secure Hash Algorithm are considered stronger than the MD5 algorithm. There have been recent advancements that may indicate a weakening of the SHA-1 variant, but nevertheless there is no evidence to suggest that the way it is used in the directory server is under any danger, nor is there any concern about any of the SHA-2 encodings.

A.4.18.16 Secure Sockets Layer

The Secure Sockets Layer (SSL) is a mechanism for wrapping network communication in a security layer that can be used to encrypt communication between the client and the server. It

also provides an integrity mechanism to ensure that the communication is not altered between the client and the server. The encryption is based on cryptography using certificate.

SSL was originally a proprietary protocol developed by Netscape Communications. It has since been standardized, but the name has been changed to Transport Security Layer. Nevertheless, SSL is still a commonly-used term to refer to this capability, and it is the term used throughout the directory server to avoid confusion with the StartTLS extended operation.

A.4.18.17 server-side sort control

The server-side sort control is a type of control that can be attached to a search operation to request that the results be sorted before they are returned to the client. It is defined in RFC 2891 (http://www.ietf.org/rfc/rfc2891.txt).

The request control has an object identifier of 1.2.840.113556.1.4.473 and the value is encoded as follows:

```
SortKeyList ::= SEQUENCE OF SEQUENCE {
           attributeType   AttributeDescription,
           orderingRule    [0] MatchingRuleId OPTIONAL,
           reverseOrder    [1] BOOLEAN DEFAULT FALSE }
```

For an example of using this control in a search request, see Searching Using the Server-Side Sort Control.

The response control has an OID of 1.2.840.113556.1.4.474 and its value is encoded as follows:

```
SortResult ::= SEQUENCE {
     sortResult  ENUMERATED {
          success                   (0), -- results are sorted
          operationsError           (1), -- server internal failure
          timeLimitExceeded         (3), -- timelimit reached before
                                         -- sorting was completed
          strongAuthRequired        (8), -- refused to return sorted
                                        -- results via insecure
                                         -- protocol
          adminLimitExceeded       (11), -- too many matching entries
                                         -- for the server to sort
          noSuchAttribute          (16), -- unrecognized attribute
                                         -- type in sort key
          inappropriateMatching    (18), -- unrecognized or
                                         -- inappropriate matching
                                         -- rule in sort key
          insufficientAccessRights (50), -- refused to return sorted
                                         -- results to this client
          busy                     (51), -- too busy to process
          unwillingToPerform       (53), -- unable to sort
          other                    (80)
          },
     attributeType [0] AttributeDescription OPTIONAL }
```

A.4.18.18 simple authentication

Simple authentication is the process of authentication to the Directory Server using a distinguished name and password. This is done using an bind operation (and when the bind is performed using simple authentication, it is often called a "simple bind"). The client uses the provided DN to identify itself to the server, and the password is used to verify that the client is who it claims to be.

Be aware that simple authentication does not protect the password in any way, and therefore it is generally recommended that it only be used over a secure communication channel like that provided by Secure Sockets Layer or StartTLS extended operation.

A.4.18.19 Simple Authentication and Security Layer

The Simple Authentication and Security Layer (SASL) is an extensible framework that is primarily used for authentication users, but in some cases it may also be used for protecting the underlying communication channel. The core functionality of SASL is described in RFC 4422 (`http://www.ietf.org/rfc/rfc4422.txt`), but some SASL mechanisms are described in other specifications.

The SASL mechanisms supported by the directory server include:

- ANONYMOUS SASL mechanism

  This mechanism does not actually authenticate users to the server, but can be used to destroy a previous authentication session.

- CRAM-MD5 SASL mechanism

  This mechanism provides a way for users to authenticate to the server using a password in a manner that does not expose the password itself. It is similar to, but weaker than, the DIGEST-MD5 SASL mechanism, and does not provide any way for ensuring connection integrity or confidentiality.

- DIGEST-MD5 SASL mechanism

  This mechanism provides a way for users to authenticate to the server using a password in a manner that does not expose the password itself. It is similar to, but stronger than, the CRAM-MD5 SASL mechanism, and also provides a way to ensure connection integrity and confidentiality.

- EXTERNAL SASL mechanism

  This mechanism provides a way for users to authenticate to the server using information available outside of the LDAP communication that has been performed (for example, the certificate that a client presented when performing Secure Sockets Layer or StartTLS extended operation negotiation).

- GSSAPI SASL mechanism

  This mechanism provides a way for users to authenticate to the server using a Kerberos V5 session. It also provides a mechanism that can be used to ensure connection integrity and confidentiality.

- PLAIN SASL mechanism

  This mechanism provides a way for users to authenticate to the server with a username and password. It is similar to the protection offered by simple authentication, but may be more convenient in that users can identify themselves with a username rather than a distinguished name.

A.4.18.20 simple paged results control

The simple paged results control is a type of control that can be attached to a search operation to indicate that only a subset of the results should be returned. It may be used to iterate through the search results a page at a time. It is similar to the virtual list view control except that it does not require the results to be sorted and can only be used to iterate sequentially through the search results.

The simple paged results control is defined in RFC 2696 (http://www.ietf.org/rfc/rfc2696.txt). The same control is used in both the search request and search result done messages. It has an object identifier of 1.2.840.113556.1.4.319, and the value is encoded as follows:

```
realSearchControlValue ::= SEQUENCE {
     size            INTEGER (0..maxInt),
                             -- requested page size from client
                             -- result set size estimate from server
     cookie          OCTET STRING
}
```

For an example of using this control in a search request, see Searching Using the Simple Paged Results Control.

A.4.18.21 size limit

The server size limit is a configuration option that controls the maximum number of entries that may be returned from any single search operation. This is a server-wide setting and may be overridden by a per-user configuration in the `ds-rlim-size-limit` operational attribute in the user's entry.

The server size limit (or per-user value) may also be restricted by the size limit element in the search request message.

A.4.18.22 smart referral

A smart referral is a special type of entry that can be placed in the directory information tree (DIT) that references content in another server, DIT location, or both. Smart referral entries contain the `referral` object class with one or more instances of the `ref` attribute containing LDAP URL that should be used in the referral.

A.4.18.23 StartTLS extended operation

The StartTLS extended operation is a type of extended operation that can be used to initiate a Transport Security Layer-secured communication channel over an otherwise clear-text connection. It allows clients to use the same network port for both secure and insecure communication.

The StartTLS extended operation is defined in RFC 4511 (http://www.ietf.org/rfc/rfc4511.txt) and further described in RFC 4513 (http://www.ietf.org/rfc/rfc4513.txt). It uses an OID of 1.3.6.1.4.1.1466.20037 with no value. The response includes an OID of 1.3.6.1.4.1.1466.20037 (the same as the request OID) with no value.

A.4.18.24 static group

A static group is a type of group in the directory server that defines its membership by providing an explicit set of distinguished name of the entry that are members of the group.

Static groups are very well supported by external clients, but are not as scalable as dynamic group when handling large numbers of members.

A.4.18.25 structural object class

A structural object class is one of the primary object class type. A structural object class is special in that it defines the core type for any entry that contains it. An entry must have exactly

one structural class (although that structural class may inherit from other structural or abstract object class classes).

The structural object class for an entry may be used by other schema elements for defining constraints on directory data. It may be used by a name form definition to control the attributes used in the relative distinguished name for the entry, and in turn by a DIT structure rule to control the types of parent entries that it may have. The structural object class may also be used by a DIT content rule to control the set of auxiliary object class and required, allowed, and prohibited attribute type for the entry.

A.4.18.26 subentry

See LDAP Subentry.

A.4.18.27 subschema subentry

A subschema subentry is a special entry within the Directory Server that provides information about the schema elements defined in the server. Attributes in this entry include:

- `ldapSyntaxes`

  The set of attribute syntax defined in the server schema.

- matchingRules

  The set of matching rule defined in the server schema.

- `matchingRuleUse`

  The set of matching rule use defined in the server schema.

- `attributeTypes`

  The set of attribute type defined in the server schema.

- `objectClasses`

  The set of object class defined in the server schema.

- `nameForms`

  The set of name form defined in the server schema.

- `dITContentRules`

  The set of DIT content rule defined in the server schema.

- `dITStructureRules`

  The set of DIT structure rule defined in the server schema.

Be aware that all of these are operational attribute and therefore will not be returned unless explicitly requested.

Also, it is technically possible for directory servers to have multiple subschema subentries with different sets of schema definitions that govern different portions of the directory information tree (DIT). The schema that applies to any given entry may be determined by retrieving the `subschemaSubentry` virtual attribute from that entry. The directory server currently supports only a single schema, and by default publishes that schema at `cn=schema`.

A.4.18.28 substring assertion

A substring assertion is the argument provided to a substring matching rule in the process of determining whether an attribute has any attribute value that matches a given substring.

The substring assertion contains at least one component from the following set:

- Zero or one subInitial element, which must appear at the beginning of the target value.
- Zero or more subAny elements, which may appear anywhere in the middle of the value. If there are multiple subAny elements, then a matching attribute value must contain all of the subAny elements in the order they appear in the substring assertion with no overlap (i.e., no character in an attribute value can be part of two different substring assertion components). If subInitial components, subFinal components, or both are present, then none of the subAny elements may overlap with them either.
- Zero or one subFinal element, which must appear at the end of the target value.

The substring assertion is used when processing a substring search filter.

A.4.18.29 substring index

A substring index is a type of index that is used to keep track of which entries contain specific substrings. Index keys for a substring index consist of six-character substrings taken from attribute values and the corresponding values are ID list containing the entry ID of the entries containing those substrings. The attribute's substring matching rule is used to normalized value the values for the index keys, and substring indexes cannot be defined for attributes that do not contain substring matching rules.

A.4.18.30 substring search filter

A substring search filter is a type of LDAP search filter that can be used to identify entries that contain a value for a given attribute that matches a specified substring. The server will use a substring matching rule to make the determination.

The substring search filter must contain a substring assertion, which will have at least one component from the following types:

- A subInitial component, whose value should be contained at the start of any matching value. There may be either zero or one subInitial component in a substring filter.
- A set of subAny components, whose values should be contained anywhere in the matching value. There may be zero or more subAny components in a substring filter, and they should be contained in the value in the order they appear in the substring filter, after any subInitial component and before any subFinal component.
- A subFinal component, whose value should be contained at the end of a matching value. There may be either zero or one subFinal component in a substring filter.

The string representation of an LDAP substring filter comprises an opening parenthesis followed by the attribute name, an equal sign, the substring assertion with the individual components separated by asterisks, and the closing parenthesis. For example, a substring filter of `(cn=ab*def*mno*stu*yz)` contains a subInitial component of `ab`, subAny components of `def`, `mno`, and `stu`, and a subFinal component of `yz`.

A.4.18.31 subtree

There are two definitions for the term "subtree".

The general definition for the term is simply a portion of the directory information tree (DIT), including an entry and all of its subordinates.

The term subtree is also described in RFC 3672 (`http://www.ietf.org/rfc/rfc3672.txt`) in the form of a subtree specification. A subtree specification provides a mechanism for grouping entries based on a given set of criteria.

A.4.18.32 subtree delete control

The subtree delete control is a type of control that can be attached to a delete operation that will allow the entry and all of its subordinate entries to be deleted. Normal delete operations may target only leaf entry, but the subtree delete control may be used to target non-leaf entry.

The subtree delete request control has an OID of `1.2.840.113556.1.4.805` with no value. There is no corresponding response control.

The following example shows the use of this control to delete the `ou=People,dc=example,dc=com` subtree.

```
$ ldapdelete -p 1389 -h localhost -D cn=directory manager -j pwd-file \
  -J 1.2.840.113556.1.4.805
ou=People,dc=example,dc=com
Processing DELETE request for ou=People,dc=example,dc=com
```

A.4.18.33 supported control

A supported control is a mechanism for identifying the request control supported by the Directory Server. The object identifier of these controls are listed in the `supportedControl` attribute of the server's root DSE.

For a list of all controls currently supported in Oracle Unified Directory, see Supported LDAP Controls.

A.4.18.34 supported extension

A supported extension is a mechanism for identifying the extended operation supported by the Directory Server. The object identifier of these extended operations are listed in the `supportedExtension` attribute of the server's root DSE.

For a list of all supported extensions for the directory server, see Supported Extended Operations.

A.4.18.35 supported feature

A supported feature is a mechanism for identifying optional capabilities that the Directory Server supports. Some features that are supported by the server are listed in the `supportedFeatures` attribute of the server's root DSE, which lists the object identifier of the supported features.

Some supported features for the directory server include:

- 1.3.6.1.4.1.4203.1.5.1

  Indicates that the server supports the use of the `+` indicator when requesting all operational attribute as specified in RFC 3673 (`http://www.ietf.org/rfc/rfc3673.txt`).

- 1.3.6.1.4.1.4203.1.5.2

  Indicates that the server supports the ability to include one or more object class names in the set of search attributes as specified in RFC 4529 (`http://www.ietf.org/rfc/rfc4529.txt`).

- 1.3.6.1.1.14

Indicates that the server supports the increment modification type, which is part of the increment modify extension as described in RFC 4525 (http://www.ietf.org/rfc/rfc4525.txt).

- 1.3.6.1.4.1.4203.1.5.3

  Indicates that the server supports LDAP true filter and LDAP false filter as described in RFC 4526 (http://www.ietf.org/rfc/rfc4526.txt).

A.4.18.36 synchronization

Data synchronization is a mechanism for keeping track of changes in the directory environment and allowing them to be reflected elsewhere.

Synchronization differs from replication in that it can occur between different vendor products, such as Active Directory and Oracle Unified Directory.

A.4.19 T

A.4.19.1 task

A task provides a set of logic for performing some type of processing in the server. Tasks are generally used to perform administrative functions within the server. Examples of tasks available for use include:

- Adding a new file to the server schema
- backup up the contents of a server back end
- restore a previous backup
- Performing an LDIF import operation
- Performing an LDIF export operation
- Initializing a replica in the server replication environment
- Performing an in-core restart
- Performing a server shutdown

Tasks can be recurring, that is scheduled to execute at regular intervals according to a specific schedule. For example, backup tasks can be made recurring to back up the server data on a regular basis. For information about scheduling tasks, see Scheduling and Configuring Tasks.

A.4.19.2 time limit

The server time limit is a configuration option that controls the maximum length of time in seconds that the server may spend processing a search operation. This is a server-wide setting and may be overridden by a per-user configuration in the `ds-rlim-time-limit` operational attribute in the user's entry.

The server time limit (or per-user value) may also be restricted by the time limit element in the search request message.

A.4.19.3 transaction

A transaction is a collection of one or more read, write, or read and write operations that occur within a database. Transactions may be described by the acronym ACID, which stands for atomicity, consistency, isolation, and durability. The directory server uses transactions in the Berkeley DB Java Edition to ensure that multiple changes made as part of a single LDAP operation (for example, updates to both the id2entry database and to index).

Even though the Directory Server uses transactions internally for its operations in the database, it does not currently expose a transactional mechanism that allows clients to perform several operations as a single atomic unit. There is an Internet Draft that describes a potential mechanism for exposing transactions (draft-zeilenga-ldap-txn), but the directory server does not currently support this capability.

A.4.19.4 Transport Security Layer

The Transport Security Layer (TLS) is a mechanism for securing network communication between clients and servers. It is the name given to the standardized form of the Secure Sockets Layer.

In most cases, the term "SSL" is preferred over "TLS" because it is the more popular term, and also to avoid confusion with the StartTLS extended operation.

A.4.19.5 true filter

See LDAP true filter

A.4.19.6 trust manager provider

A trust manager provider is a component of the server that can provide information that can be used to determine whether to trust certificates presented to the server.

A.4.19.7 typesOnly flag

The TypesOnly flag is an element of an search operation that indicates whether attributes returned as part of search result entry should include only the attribute description or both the attribute description and the attribute value.

A.4.20 U

A.4.20.1 unbind operation

The LDAP unbind operation is used to indicate that the client wants to disconnect from the server.

> **Note:**
>
> The unbind operation cannot be used to destroy an authentication session while leaving the underlying connection established. If the client does not close the connection after sending an unbind request, then the server will. If there is a need to revert a connection to an unauthenticated state, then you should perform an anonymous bind operation.

The LDAP unbind request protocol op is defined as follows:

```
UnbindRequest ::= [APPLICATION 2] NULL
```

An unbind request does not contain any elements, and the server will not send a response to an unbind request.

A.4.20.2 unindexed search

An unindexed search is one that cannot be processed using the set of index defined in the server. It will necessitate iterating through most or all of the entries in the database.

Unindexed searches can be expensive for the server to process, users will only be allowed to perform unindexed searches if they have the `unindexed-search` privilege.

For more information, see Indexing Directory Data.

A.4.20.3 UNIX crypt algorithm

The UNIX crypt algorithm is a standard mechanism for encoding user passwords using a DES-based encryption scheme that ultimately results in a one-way message digest. It is called the "UNIX crypt" algorithm because it has historically been used as the default mechanism for encoding passwords in UNIX-based systems.

The UNIX crypt algorithm is considered weak because it is based on a 56-bit encryption algorithm and uses only a 12-bit salt. Therefore, it should only be used in cases where clients expect to be able to retrieve the password from the server and compare its value against what the user supplied instead of attempting to verify it using an bind operation.

See also crypt algorithm.

A.4.20.4 unsolicited notification

An unsolicited notification is a type of extended operation message that is special in that the server generates this kind of message without any corresponding request from the client. It may be used to notify the client of some important information.

The directory server currently supports a single unsolicited notification: the notice of disconnection unsolicited notification, which can be used to inform the client that the server is closing the connection.

A.4.20.5 URL

See URL.

A.4.20.6 user attribute

A user attribute is an attribute type with an attribute usage of `userApplications`. User attributes are used for actually storing information in the directory, as opposed to operational attribute which are used for storing state information used for internal server processing.

Whenever a search operation does not request any specific attributes to be returned, then all user attributes in matching entries will be returned. An explicit value of `*` (the asterisk) may also be included to explicitly include all user attributes.

A.4.21 V

A.4.21.1 virtual attribute

A virtual attribute is a type of attribute in which the attribute value is not actually stored in the back end but is dynamically generated in some manner. The values can be obtained in various ways, depending on the type of virtual attribute. Some virtual attributes use a hard-coded value, while others compute their values at runtime based on some kind of logic.

A.4.21.2 virtual attributes only control

The virtual attributes only control requests that the server include only virtual attribute in matching entries. That is, real attributes are excluded from search result entry.

The virtual attributes only control has a request object identifier of `2.16.840.1.113730.3.4.19` and no value.

The following example shows a search on the base DN without the virtual attributes only control:

```
$ ldapsearch -p 1389 -D "cn=directory manager" -j pwd-file -b "dc=example,dc=com" \
  -s base "objectclass=*"
version: 1
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

The following example shows the same search with the virtual attributes only control:

```
$ ldapsearch -p 1389 -D "cn=directory manager" -j pwd-file \
  -J "2.16.840.1.113730.3.4.19"   -b "dc=example,dc=com" -s base "objectclass=*"
version: 1
dn: dc=example,dc=com
```

A.4.21.3 virtual directory

A virtual directory is a type of network daemon that communicates with clients using Lightweight Directory Access Protocol but obtains the underlying data from a combination of different sources. Virtual directories may have several different capabilities, including:

- Providing an LDAP front end to a different repository, like a relational database or a flat file
- Providing a mechanism to merge data from multiple repositories

A.4.21.4 virtual list view control

The virtual list view (VLV) control can be attached to a search operation to indicate that only a subset of the results are to be returned. It can be used to iterate through the search results a page at a time. It is similar to the simple paged results control except that it can be used to retrieve an arbitrary subset of the results from the server, and it requires that the search request also include the server-side sort control to ensure that the results are consistently sorted across requests.

The VLV control is defined in draft-ietf-ldapext-ldapv3-vlv-09 (http://tools.ietf.org/html/draft-ietf-ldapext-ldapv3-vlv-09). The request control has an object identifier of 2.16.840.1.113730.3.4.9 and the value is encoded as follows:

```
VirtualListViewRequest ::= SEQUENCE {
     beforeCount    INTEGER (0..maxInt),
     afterCount     INTEGER (0..maxInt),
     target       CHOICE {
          byOffset        [0] SEQUENCE {
               offset          INTEGER (1 .. maxInt),
               contentCount    INTEGER (0 .. maxInt) },
          greaterThanOrEqual [1] AssertionValue },
     contextID     OCTET STRING OPTIONAL }
```

The response control has an OID of 2.16.840.1.113730.3.4.10 and the value is encoded as shown below:

```
VirtualListViewResponse ::= SEQUENCE {
     targetPosition    INTEGER (0 .. maxInt),
     contentCount     INTEGER (0 .. maxInt),
     virtualListViewResult ENUMERATED {
          success (0),
          operationsError (1),
          protocolError (3),
          unwillingToPerform (53),
          insufficientAccessRights (50),
          timeLimitExceeded (3),
          adminLimitExceeded (11),
          innapropriateMatching (18),
          sortControlMissing (60),
          offsetRangeError (61),
         other(80),
          ... },
     contextID     OCTET STRING OPTIONAL }
```

For an example of using this control in a search request, see Searching Using the Virtual List View Control.

A.4.21.5 virtual static group

A virtual static group is a special type of group that appears to be static group to external clients but obtains its membership information from another group (like a dynamic group) in the server.

Virtual static groups are primarily used in cases where a client application only supports static groups but have a very large number of members that are better suited for maintaining in a dynamic group.

A.4.21.6 VLV index

A virtual list view (VLV) index is a mechanism used by the Directory Server database that can be used to efficiently process searches with virtual list view control. A VLV index effectively notifies the server that a virtual list view, with specific query and sort parameters, will be performed. This index also allows the server to collect and maintain the information required to make using the virtual list view faster. A VLV index stores sorted blocks of ID list, which are a set of entry ID and the attribute values of the entry to sort on.

A.4.22 W

A.4.22.1 "Who Am I?" extended operation

The "Who Am I?" extended operation provides an extended operation for determining the authorization identity of a client connection. It is defined in RFC 4532 (`http://www.ietf.org/rfc/rfc4532.txt`).

The request object identifier for the "Who Am I?" extended operation is `1.3.6.1.4.1.4203.1.11.3`, and there should not be a request value. The response should not include a response OID, and the value should be a string containing the client's authorization identity (or it may be an empty string if the authorization identity is that of the anonymous user).

The information provided by the "Who Am I?" extended operation is similar to that provided by the authorization identity control except that it can be used at any time after the client has authenticated, whereas the authorization identity control can only be included with a bind request.

A.4.22.2 work queue

The Directory Server work queue is the mechanism that it uses to keep track of outstanding requests and ensuring that they are processed in an appropriate manner. The work queue functionality is provided by an extensible API, but the default implementation is relatively simple: a queue is serviced by several worker thread. If there are free worker threads, then the queue will generally remain empty. If all worker threads are busy, then subsequent requests will be placed in the work queue so that they are processed in a FIFO manner.

A.4.22.3 worker thread

A worker thread is a thread used to process requests in the Directory Server. Worker threads are associated with the work queue, and they will operate in a loop that includes picking up a request from the queue (waiting for a request to arrive if necessary), processing that request appropriate, and then returning to the queue for the next request.

A.4.22.4 workflow

A workflow defines the processing for a given naming context. The overall processing is split into a set of ordered and synchronized tasks, defined by *workflow elements*.

A.4.22.5 workflow element

A workflow element is the key building block of a workflow processing. It defines how the client request sent to the server will be treated. The workflow elements implement the main tasks in the proxy server, including for example, load balancing and distribution.

A.4.22.6 writability mode

The writability mode of the Directory Server is used to control whether write operations are allowed. The writability mode configuration can be restricted to a single back end or it can apply to the entire server.

The following writability modes are available:

- enabled

  The server attempts to process all write operations

- disabled

  The server rejects all write operations

- internal-only

  The server attempts to process write operations initiated as internal operations or through synchronization but rejects any request coming from an external client

An entryDN is an operational attribute that provides a copy of the entry's current DN. Because a DN is not an attribute of the entry, it cannot be used to perform attribute value assertions. The entryDN provides a mechanism to access an entry's DN and is described in RFC 5020 (http://www.ietf.org/rfc/rfc5020.txt).