Oracle® Fusion Middleware

Administering Oracle Internet Directory

14c (14.1.2.1.0)
G10426-01
March 2025

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

G10426-01

Copyright © 2011, 2025, Oracle and/or its affiliates.

Primary Author: Devanshi Mohan

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Audience xliii
Documentation Accessibility xliii
Related Documents xliii
Conventions xliv

What's New in Oracle Internet Directory?

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

Part I Understanding Directory Services

1 Introduction to Directory Services

1.1 What is a Directory? 1-1
1.2 Understanding Directory and its Role 1-1
1.2.1 The Expanding Role of Online Directories 1-1
1.2.2 The Problem: Too Many Special-Purpose Directories 1-2
1.3 What is Lightweight Directory Access Protocol (LDAP)? 1-3
1.3.1 LDAP and Simplified Directory Management 1-3
1.3.2 LDAP Version 3 1-3
1.4 Understanding Oracle Internet Directory 1-4
1.4.1 Overview of Oracle Internet Directory 1-4
1.4.2 Components of Oracle Internet Directory 1-5
1.4.3 Advantages of Oracle Internet Directory 1-6
1.4.3.1 Scalability in Oracle Internet Directory 1-6
1.4.3.2 High Availability in Oracle Internet Directory 1-6
1.4.3.3 Security Benefits in Oracle Internet Directory 1-7
1.4.3.4 Integration of Oracle Internet Directory with Oracle Environment 1-7
1.5 How Oracle Products Use Oracle Internet Directory 1-7
1.5.1 Easier and More Cost-Effective Administration of Oracle Products 1-7
1.5.2 Tighter Security Through Centralized Security Policy Administration in Oracle Internet Directory 1-8

1.5.3 Integration of Multiple Directories with Oracle Internet Directory 1-8

2 Understanding Oracle Internet Directory in Oracle Fusion Middleware

2.1 Understanding WebLogic Server Domain 2-1
2.2 Oracle Internet Directory as a System Component 2-1
2.3 Oracle Internet Directory Deployment Options 2-2
2.4 Middleware Home 2-2
2.5 WebLogic Server Home 2-3
2.6 Oracle Common Home 2-3
2.7 Oracle Home 2-3
2.8 Oracle Instance in 14c Release 2-3
2.9 Oracle Enterprise Manager Fusion Middleware Control 2-4
2.10 Known Limitations of Oracle Enterprise Manager for Oracle Internet Directory 2-4
2.11 Logging, Auditing and Diagnostics Using Fusion Middleware Control 2-4
2.12 MBeans and the WebLogic Scripting Tool 2-5

3 Understanding the Concepts and Architecture of Oracle Internet Directory

3.1 Understanding the Architecture of Oracle Internet Directory 3-1
3.1.1 Oracle Internet Directory Node 3-2
3.1.2 Oracle Directory Server Instance 3-4
3.1.3 Oracle Internet Directory Ports 3-5
3.1.4 Directory Metadata 3-5
3.2 Understanding How Oracle Internet Directory Processes a Search Request 3-7
3.2.1 Standard LDAP Search Operations 3-7
3.2.2 Persistent LDAP Search Operations 3-8
3.3 Understanding Directory Entries in Oracle Internet Directory 3-9
3.3.1 Distinguished Names and Directory Information Trees 3-10
3.3.2 Entry Caching 3-11
3.4 Understanding the Concept of Attributes in Oracle Internet Directory 3-11
3.4.1 Attributes in Online Directory 3-11
3.4.2 Application and System Configuration Attributes 3-12
3.4.3 Attributes Created with Each New Entry 3-13
3.4.4 Single-Valued and Multivalued Attributes 3-13
3.4.5 Common LDAP Attributes 3-13
3.4.6 Attribute Syntax 3-14
3.4.7 Attribute Matching Rules 3-14
3.4.8 Attribute Options 3-15
3.5 Understanding Object Classes in Oracle Internet Directory 3-15
3.5.1 Subclasses, Superclasses, and Inheritance 3-16
3.5.2 Understanding the Types of Object Class 3-16

3.5.2.1 Structural Object Classes 3-16
3.5.2.2 Auxiliary Object Classes 3-17
3.5.2.3 Abstract Object Classes 3-17
3.6 Directory Naming Contexts 3-17
3.7 Security Features in Oracle Internet Directory 3-18
3.8 Globalization Support 3-19
3.9 Understanding Distributed Directories 3-20
3.9.1 Directory Replication 3-20
3.9.2 Directory Partitioning 3-20
3.10 Knowledge References and Referrals 3-21
3.11 Service Registry and Service to Service Authentication 3-23
3.12 Oracle Directory Integration Platform 3-24
3.13 Understanding the Role of Identity Management in Oracle Internet Directory 3-24
3.13.1 Identity Management 3-24
3.13.2 Understanding the Identity Management Realms 3-25
3.13.2.1 Identity Management Realm 3-25
3.13.2.2 Default Identity Management Realm 3-26
3.13.2.3 Identity Management Policies 3-26
3.14 TCP Keep-Alive Mechanism 3-26
3.15 Understanding the Concept of Resource Information 3-27
3.15.1 Resource Type Information 3-27
3.15.2 Resource Access Information 3-28
3.15.3 Location of Resource Information in the DIT 3-28

4 Understanding Process Control of Oracle Internet Directory Components

4.1 Oracle Internet Directory Process Control Architecture 4-1
4.2 The ODS_PROCESS_STATUS Table in Oracle Internet Directory 4-2
4.3 Starting, Stopping, and Monitoring of Oracle Internet Directory Processes 4-3
4.3.1 Starting Oracle Internet Directory Using WLST Command 4-3
4.3.2 Shutting Down Oracle Internet Directory Using WLST Command — shutdown () 4-4
4.3.3 Server Process Monitoring Using Node Manager 4-5
4.4 Oracle Internet Directory Replication-Server Control and Failover 4-5
4.4.1 Understanding the OID Monitor and Replication Server Failover 4-5
4.4.2 Enabling Failover for Oracle Internet Directory Processes 4-6
4.5 Oracle Internet Directory Process Control: Best Practices 4-6

5 Understanding Oracle Internet Directory Organization

5.1 Understanding Directory Information Tree 5-1
5.2 How to Plan the Overall Directory Structure 5-2
5.3 Planning the Names and Organizing Users and Groups 5-3

5.3.1 Organizing Users in Oracle Internet Directory 5-4
5.3.2 Organizing Groups in Oracle Internet Directory 5-5
5.4 About DIT View Since 11g Release 5-5
5.5 Migration of DIT from a Third-Party Directory 5-6

6 Understanding Oracle Internet Directory Replication

6.1 What is Oracle Internet Directory Replication? 6-1
6.2 Why Use Oracle Internet Directory Replication? 6-2
6.3 Understanding Basic Concepts of Internet Directory Replication 6-2
6.3.1 How to Decide Full or Partial Content Replication? 6-2
6.3.2 Replication Direction: One-Way, Two-Way, or Peer to Peer 6-3
6.3.3 Transport Mechanism: LDAP 6-4
6.3.4 Directory Replication Group (DRG) Type: Single-master, Multimaster, or Fan-out 6-4
6.3.5 Example for Single-master, Multimaster, or Fan-out Directory Replication Group Type 6-5
6.3.5.1 Single-Master Replication Example 6-5
6.3.5.2 Multimaster Replication Example 6-6
6.3.5.3 Fan-out Replication Example 6-6
6.3.6 Loose Consistency Model in Directory Replication Architecture 6-7
6.3.7 Multimaster Replication with Fan-Out 6-7
6.4 What Kind of Replication Do You Need? 6-8

Part II Basic Administration

7 Getting Started With Oracle Internet Directory

7.1 Overview of Postinstallation Tasks and Information 7-1
7.1.1 Setting Up the Environment 7-2
7.1.2 Adding Datafiles to the OLTS_CT_STORE and OLTS_ATTRSTORE Tablespaces 7-2
7.1.3 Changing Settings of Windows Services 7-2
7.1.4 Starting and Stopping the Oracle Stack 7-2
7.1.5 Default URLs and Ports 7-2
7.1.6 About Tuning Oracle Internet Directory 7-2
7.1.7 Enabling Anonymous Binds 7-3
7.1.8 Enabling Oracle Internet Directory to run on Privileged Ports 7-3
7.1.9 Verifying Oracle Database Time Zone 7-4
7.2 Overview of Using Fusion Middleware Control to Manage Oracle Internet Directory 7-4
7.2.1 Managing Oracle Internet Directory Using Fusion Middleware Control 7-5
7.2.2 Oracle Internet Directory Menu 7-5
7.3 Overview of Oracle Directory Services Manager 7-6

7.3.1 Understanding Oracle Directory Services Manager 7-7
7.3.1.1 About the JAWS Screen Reader with Oracle Directory Services Manager 7-7
7.3.1.2 Non-Super User Access to Oracle Directory Services Manager 7-7
7.3.1.3 Single Sign-On Integration with Oracle Directory Services Manager 7-7
7.3.2 Configuring ODSM for SSO Integration 7-8
7.3.3 Configuring the SSO Server for ODSM Integration 7-9
7.3.4 About Configuring the Oracle HTTP Server for ODSM-SSO Integration 7-9
7.3.5 Invoking Oracle Directory Services Manager 7-10
7.3.6 Overview of Connecting to the Server from Oracle Directory Services Manager 7-10
7.3.6.1 Logging into the Directory Server from Oracle Directory Services Manager 7-11
7.3.6.2 Understanding Logging Into the Directory Server from Oracle Directory Services Manager Using SSL 7-12
7.3.6.3 Connecting to an SSO-Enabled Directory as an SSO-Authenticated User 7-13
7.3.7 Configuring Oracle Directory Services Manager Session Timeout 7-13
7.3.8 Configuring Oracle HTTP Server to Support Oracle Directory Services Manager in an Oracle WebLogic Server Cluster 7-14
7.4 Overview of Managing Oracle Internet Directory Using Command-Line Utilities 7-15
7.4.1 About Setting Environmental Variables to Use Oracle Internet Directory Command-Line Utilities 7-15
7.4.2 About Standard LDAP Utilities 7-15
7.4.3 Bulk Tools 7-16
7.4.4 About WLST 7-16
7.5 Basic Tasks for Configuring and Managing Oracle Internet Directory 7-17

8 Managing Oracle Internet Directory Instances

8.1 Overview of Managing Oracle Internet Directory Instances 8-1
8.1.1 About the Instance-Specific Configuration Entry 8-1
8.1.2 About the First Oracle Internet Directory Instance Creation 8-2
8.1.3 Creating Additional Oracle Internet Directory Instances 8-3
8.1.4 Registering an Oracle Instance or Component with the WebLogic Server 8-4
8.2 Overview of Oracle Internet Directory Components Management by Using Fusion Middleware Control 8-5
8.2.1 Viewing Active Server Information by Using Fusion Middleware Control 8-5
8.2.2 Starting the Oracle Internet Directory Server by Using Fusion Middleware Control 8-5
8.2.3 Stopping the Oracle Internet Directory Server by Using Fusion Middleware Control 8-6
8.2.4 Restarting the Oracle Internet Directory Server by Using Fusion Middleware Control 8-6
8.3 Managing Oracle Internet Directory Components by Using WLST Commands 8-6
8.3.1 Creating an Oracle Internet Directory Component by Using WLST Command — oid_createInstance 8-7

8.3.2 Deleting an Oracle Internet Directory Component by Using WLST Command — oid_deleteInstance() 8-8
8.3.3 Viewing Active Server Instance Information by Using WLST Command — oid_instanceStatus() 8-8
8.3.4 Starting the Oracle Internet Directory Server by Using WLST Command — start() 8-10
8.3.5 Stopping the Oracle Internet Directory Server by Using WLST Command — shutdown() 8-11
8.3.6 Updating Credential Required by Enterprise Manager to manage OID - oid_setProperties() 8-11
8.3.7 Fetching Enterprise Manager Properties Used to Manage OID - oid_getProperties() 8-12
8.3.8 Creating a Realm in Oracle Internet Directory - oid_createRealm() 8-13
8.3.9 Listing all Oracle Internet Directory Instance Names - oid_listInstances() 8-13
8.3.10 Updating Orcladmin Password - oid_setAdminPassword() 8-14
8.4 Starting an Instance of the Replication Server by Using OIDCTL 8-14

9 Managing System Configuration Attributes

9.1 Managing System Configuration Attributes 9-1
9.1.1 About Configuration Attributes 9-1
9.1.2 About Operational Attributes 9-2
9.1.3 Attributes of the Instance-Specific Configuration Entry 9-3
9.1.4 Attributes of the DSA Configuration Entry 9-9
9.1.5 Attributes of the DSE 9-14
9.2 Managing System Configuration Attributes by Using Fusion Middleware Control 9-15
9.2.1 Configuring Server Properties 9-15
9.2.1.1 Configuring Server Properties 9-16
9.2.1.2 General Options in Configuring Server Properties 9-16
9.2.1.3 Performance Options in Configuring Server Properties 9-16
9.2.1.4 SASL Tab of Server Properties 9-17
9.2.1.5 Statistics Tab of Server Properties 9-17
9.2.1.6 Logging Tab of Server Properties 9-17
9.2.2 Configuring Shared Properties 9-17
9.2.2.1 Configuring Shared Properties 9-17
9.2.2.2 Configuration Attributes in General Properties 9-18
9.2.2.3 Change Superuser Password 9-18
9.2.2.4 Replication 9-18
9.2.3 SSL and Audit Parameters Configuration 9-18
9.3 Managing System Configuration Attributes by Using WLST 9-19
9.3.1 Managing System Configuration Attributes Using WLST 9-19
9.3.2 Related MBeans Of Oracle Internet Directory 9-21
9.4 Managing System Configuration Attributes by Using LDAP Tools 9-21
9.4.1 Setting System Configuration Attributes by Using ldapmodify 9-21

9.4.2 Listing Configuration Attributes with ldapsearch 9-22
9.5 Managing System Configuration Attributes by Using ODSM Data Browser 9-23
9.5.1 Navigating to the Instance-Specific Configuration Entry 9-23
9.5.2 Navigating to the DSA Configuration Entry 9-23
9.5.3 Navigating to the DSE Root 9-24

10 Managing IP Addresses in Oracle Internet Directory

10.1 Introduction to Managing IP Addresses 10-1
10.2 Configuring an IP Address for IPV6, Cold Failover Cluster, or Virtual IP 10-1
10.3 Configuring IP Addresses for Notifications in a Cluster 10-2
10.3.1 Configuring a Dedicated IP Address and Oracle Internet Directory Instance for Notifications 10-2
10.3.2 Configuring an Oracle Internet Directory Instance with an IP Address 10-2
10.3.3 Configuring an Oracle Internet Directory Instance with a Port Number 10-3

11 Managing Naming Contexts in Oracle Internet Directory

11.1 Publishing Naming Contexts 11-1
11.2 Searching for Published Naming Contexts 11-1
11.3 Modifying a Naming Context 11-2

12 Managing Accounts and Passwords in Oracle Internet Directory

12.1 Introduction to Managing Accounts and Passwords 12-1
12.2 Managing Accounts and Passwords by Using Command-Line Tools 12-2
12.2.1 Enabling and Disabling Accounts by Using Command-Line Tools 12-3
12.2.2 Unlocking Accounts by Using Command-Line Tools 12-3
12.2.3 Forcing a Password Change by Using Command-Line Tools 12-3
12.3 Managing Accounts and Passwords by Using the Self-Service Console 12-4
12.3.1 Enabling and Disabling Accounts by Using the Oracle Internet Directory Self-Service Console 12-4
12.3.2 Unlocking Accounts by Using the Oracle Internet Directory Self-Service Console 12-5
12.3.3 Resetting Your Own Password by Using the Oracle Internet Directory Self-Service Console 12-5
12.4 Unlocking Locked Accounts by Using Oracle Directory Services Manager 12-5
12.5 Changing the Superuser Password by Using Fusion Middleware Control 12-6
12.6 Creating Another Account With Superuser Privileges 12-6
12.7 Managing the Superuser Password by Using ldapmodify 12-7
12.8 Changing the Oracle Internet Directory Database Password 12-7
12.9 Resetting the Superuser Password 12-8
12.10 Changing the Password for the EMD Administrator Account 12-8

12.11 Changing the Password for the ODSSM Administrator Account 12-9
12.12 Updating the New ODSSM Password for Data Source 12-10

13 Managing Directory Entries in Oracle Internet Directory

13.1 Introduction to Managing Directory Entries 13-1
13.2 Managing Entries by Using Oracle Directory Services Manager 13-1
13.2.1 Displaying Entries by Using Oracle Directory Services Manager 13-2
13.2.2 Searching for Entries by Using Oracle Directory Services Manager 13-4
13.2.3 Importing Entries from an LDIF File by Using Oracle Directory Services Manager 13-6
13.2.4 Exporting Entries to an LDIF File by Using Oracle Directory Services Manager 13-6
13.2.5 Viewing Attributes for a Specific Entry by Using Oracle Directory Services Manager 13-6
13.2.6 Adding a New Entry by Using Oracle Directory Services Manager 13-7
13.2.7 Deleting an Entry or Subtree by Using Oracle Directory Services Manager 13-8
13.2.8 Adding an Entry by Copying an Existing Entry in Oracle Directory Services Manager 13-8
13.2.9 Modifying an Entry by Using Oracle Directory Services Manager 13-10
13.3 Managing Entries by Using LDAP Command-Line Tools 13-11
13.3.1 Listing All the Attributes in the Directory by Using ldapsearch 13-11
13.3.2 Listing Operational Attributes by Using ldapsearch 13-12
13.3.3 Changing the Attribute Case in ldapsearch Output 13-12
13.3.4 Adding a User Entry by Using ldapadd 13-12
13.3.5 Modifying a User Entry by Using ldapmodify 13-13
13.3.6 Adding an Attribute Option by Using ldapmodify 13-14
13.3.7 Deleting an Attribute Option by Using ldapmodify 13-14
13.3.8 Searching for Entries with Attribute Options by Using ldapsearch 13-14

14 Managing Dynamic and Static Groups in Oracle Internet Directory

14.1 Understanding Dynamic and Static Groups 14-1
14.1.1 Defining Static Groups 14-2
14.1.1.1 Static Group 14-2
14.1.1.2 Schema Elements for Creating a Static Group 14-2
14.1.2 Defining Dynamic Groups 14-2
14.1.2.1 Dynamic Group 14-2
14.1.2.2 Cached and Uncached Dynamic Groups 14-3
14.1.2.3 Enhancements of Dynamic Groups in Oracle Internet Directory 14-4
14.1.2.4 Limitations of Dynamic Groups in Oracle Internet Directory 14-4
14.1.2.5 Schema Elements for Creating a Dynamic Group 14-5
14.1.2.6 About labeledURI Attribute 14-5
14.1.2.7 About CONNECT BY Assertion 14-6

14.1.2.8 Example of a Dynamic Group Entry Using the labeledURI Attribute 14-7
14.1.2.9 Example of a Dynamic List Entry Using the labeledURI Attribute 14-7
14.1.2.10 Example of a Dynamic Group Entry Using the CONNECT BY Assertion 14-7
14.1.3 About Hierarchies of Group Entries 14-8
14.1.4 About Querying Group Entries 14-8
14.1.5 Understanding the orclMemberOf Attribute 14-9
14.1.5.1 About orclMemberOf Attribute 14-9
14.1.5.2 Examples of Using the orclMemberOf Attribute 14-10
14.1.6 Considerations for Using Static and Dynamic Group 14-11
14.2 Managing Group Entries by Using Oracle Directory Services Manager 14-11
14.2.1 Creating Static Group Entries by Using Oracle Directory Services Manager 14-12
14.2.2 Adding an Owner or Member to a Static Group Entry 14-12
14.2.3 Modifying an Attribute of a Static Group Entry 14-13
14.2.4 Creating Dynamic Group Entries by Using Oracle Directory Services Manager 14-14
14.2.5 Adding an Owner or Member to a Dynamic Group Entry 14-15
14.2.6 Modifying an Attribute of a Dynamic Group Entry 14-15
14.2.7 Modifying a Dynamic Group Entry by Using Oracle Directory Services Manager 14-16
14.3 Managing Group Entries by Using the Command Line 14-17
14.3.1 Creating a Static Group Entry by Using ldapadd 14-17
14.3.2 Modifying a Static Group by Using ldapmodify 14-18
14.3.3 Creating a Dynamic Group Entry by Using ldapadd 14-18
14.3.3.1 Creating a Cached Dynamic Group Using labeledURI Attribute 14-19
14.3.3.2 Creating an Uncached Dynamic List Using labeledURI Attribute 14-19
14.3.3.3 Creating a Dynamic Group Using CONNECT BY String 14-19
14.3.4 Modifying a Dynamic Group by Using ldapmodify 14-20

15 Performing Bulk Operations

15.1 Introduction to Performing Bulk Operations 15-1
15.2 Setting Environment Variables Before Using Command-line Tools 15-2
15.3 Changing the Server Mode 15-3
15.3.1 Setting the Server Mode by Using Fusion Middleware Control 15-3
15.3.2 Setting the Server Mode by Using ldapmodify 15-3
15.4 Loading Data Into the Schema by Using bulkload 15-4
15.4.1 Different Phases of Loading Data 15-5
15.4.2 Output File Locations for bulkload Tool 15-6
15.4.3 Running the bulkload Tool 15-6
15.4.4 Importing an LDIF File by Using bulkload 15-7
15.4.4.1 Stopping Oracle Internet Directory Processes 15-7
15.4.4.2 Backing Up the Oracle Database Server 15-7
15.4.4.3 Finding Out the Oracle Internet Directory Password 15-7
15.4.4.4 Checking Input File and Generating Files for SQL*Loader 15-8

15.4.4.5 Loading Input Files 15-8
15.4.5 Loading Data in Incremental or Append Mode By Using bulkload 15-9
15.4.6 Performing Index Verification By Using bulkload 15-9
15.4.7 Re-Creating Indexes By Using bulkload 15-10
15.4.8 Recovering Data After a Load Failure By Using bulkload 15-10
15.5 Modifying Attributes By Using bulkmodify 15-10
15.5.1 Attributes Excluded from bulkmodify Operations 15-10
15.5.2 Log File Location for bulkmodify Tool 15-11
15.5.3 Running the bulkmodify Tool 15-11
15.5.4 Adding a Description for All Entries Under a Specified Naming Context 15-11
15.5.5 Adding an Attribute for Specific Entries Under a Specified Naming Context 15-12
15.5.6 Replacing an Attribute for All Entries Under a Specified Naming Context 15-12
15.6 Deleting Entries by Using bulkdelete 15-12
15.6.1 Log File Location for bulkdelete Tool 15-13
15.6.2 Running the bulkdelete Tool 15-13
15.6.3 Deleting All Entries Under a Specified Naming Context by Using bulkdelete 15-13
15.6.4 Deleting Entries Under a Naming Context and Making them Tombstone Entries 15-13
15.6.5 Deleting All Entries Under a Specified Subtree by Applying the Filter Option 15-14
15.7 Dumping Data from Oracle Internet Directory to a File by Using ldifwrite 15-14
15.7.1 Log File Location for ldifwrite Tool 15-14
15.7.2 Running the ldifwrite Tool 15-15
15.7.3 Dumping Part of a Specified Naming Context to an LDIF File 15-15
15.7.4 Dumping Entries Under a Specified Naming Context to an LDIF File 15-15
15.8 Creating and Dropping Indexes from Existing Attributes by Using catalog 15-15
15.8.1 Log File Location for catalog Tool 15-16
15.8.2 Running the catalog Tool 15-16
15.8.3 Changing a Searchable Attribute into a Non-searchable Attribute 15-17
15.8.4 Changing a Non-searchable Attribute into a Searchable Attribute 15-17

16 Managing Collective Attributes

16.1 Introduction to Collective Attributes 16-1
16.1.1 RFC Definition and Oracle Extensions 16-1
16.1.1.1 RFC 3671 16-1
16.1.1.2 Oracle Extensions to RFC 3671 16-2
16.1.2 Defining the Collective Attribute Subentry 16-2
16.1.3 Using subtreeSpecification Attribute 16-2
16.1.3.1 Limiting Collective Attribute to a Subtree by using base Keyword 16-3
16.1.3.2 Controlling Number of RDNs by using minimum and maximum Keywords 16-3
16.1.3.3 Applying Specific Exclusions by using chopBefore and chopAfter Keywords 16-3
16.1.3.4 Including Certain Object Classes by using specificationFilter 16-3

16.1.4 Overriding a Collective Attribute 16-5
16.2 Managing Collective Attributes by Using the Command Line 16-5
16.2.1 Adding a Subentry by Using ldapadd 16-5
16.2.2 Modifying a Subentry by Using ldapmodify 16-5

17 Managing Computed Attributes

17.1 Introduction to Computed Attributes 17-1
17.2 Configuring Computed Attributes 17-2
17.2.1 Rules and Syntax Used for Computed Attributes 17-2
17.2.2 Using Special Characters With Rules for Computed Attributes 17-3
17.3 Examples of Computed Attributes Using LDAP Command-Line Tools 17-3
17.3.1 Returning an Attribute Value as Uppercase 17-4
17.3.2 Returning the Substring of an Attribute Value 17-4
17.3.3 Replacing an Attribute Value 17-4
17.3.4 Specifying a URI-Based Configuration 17-5
17.3.5 Using a Combination of Different Rules 17-5
17.3.6 Using an OR (|) Operator 17-5
17.3.7 Using the connectBy Interface 17-6
17.3.8 Creating Hierarchical Groups Using connectBy 17-7

18 Managing Alias Entries

18.1 Introduction to Alias Entries 18-1
18.2 Adding an Alias Entry 18-2
18.3 Searching the Directory with Alias Entries 18-3
18.3.1 Flags for Searching the Directory with Alias Entries 18-3
18.3.2 Searching the Base with Alias Entries 18-4
18.3.2.1 Searching the Base with Dereferencing Flag -a find 18-4
18.3.2.2 Searching the Base with Dereferencing Flag -a search 18-4
18.3.2.3 Searching the Base with Dereferencing Flag -a always 18-4
18.3.3 Searching One-Level with Alias Entries 18-5
18.3.3.1 Searching One-Level with Dereferencing Flag -a find 18-5
18.3.3.2 Searching One-Level with Dereferencing Flag -a search 18-5
18.3.3.3 Searching One-Level with Dereferencing Flag -a always 18-5
18.3.4 Searching a Subtree with Alias Entries 18-6
18.3.4.1 Searching Subtree with Dereferencing Flag -a find 18-6
18.3.4.2 Searching Subtree with Dereferencing Flag -a search 18-6
18.3.4.3 Searching Subtree with Dereferencing Flag -a always 18-7
18.4 Modifying Alias Entries 18-7
18.5 Messages Related to Alias Dereferencing 18-7

19 Managing Attribute Uniqueness Constraint Entries

19.1 Introduction to Managing Attribute Uniqueness Constraint Entries 19-1
19.1.1 Attribute Uniqueness Scope 19-1
19.1.2 Attribute Uniqueness Constraint Entries 19-2
19.1.3 Attribute Uniqueness Constraint in Oracle Internet Directory Replication Environment 19-2
19.1.3.1 Attribute Uniqueness Constraint in Simple Replication Environment 19-2
19.1.3.2 Attribute Uniqueness Constraint in Multimaster Replication Environment 19-3
19.1.4 Support of LDAP Tools for Attribute Uniqueness 19-3
19.2 Duplicate Attribute Values 19-4
19.3 Cleaning Up Duplicate Attribute Values 19-4
19.4 Specifying Attribute Uniqueness Constraint Entries 19-4
19.4.1 Specifying Multiple Attribute Names in an Attribute Uniqueness Constraint 19-5
19.4.2 Specifying Multiple Subtrees in an Attribute Uniqueness Constraint 19-6
19.4.3 Specifying Multiple Scopes in an Attribute Uniqueness Constraint 19-6
19.4.4 Specifying Multiple Object Classes in an Attribute Uniqueness Constraint 19-7
19.4.5 Specifying Multiple Subtrees, Scopes, and Object Classes in an Attribute Uniqueness Constraint 19-7
19.5 Managing an Attribute Uniqueness Constraint Entry by Using ODSM 19-8
19.5.1 Creating an Attribute Uniqueness Constraint Entry by Using ODSM 19-8
19.5.2 Modifying an Attribute Uniqueness Constraint Entry by Using ODSM 19-8
19.5.3 Deleting an Attribute Uniqueness Constraint Entry by Using ODSM 19-9
19.6 Managing Attribute Uniqueness Constraint Entries by Using the Command Line 19-9
19.6.1 Creating Attribute Uniqueness Across a Directory by Using Command-Line 19-9
19.6.2 Specifying Uniqueness Constraint for an Attribute by Using Command-Line 19-10
19.6.3 Creating Attribute Uniqueness Across One Subtree by Using Command-Line 19-10
19.6.4 Creating Attribute Uniqueness Across One Object Class by Using Command-Line 19-11
19.6.5 Modifying Attribute Uniqueness Constraint Entries by Using Command-Line 19-11
19.6.6 Deleting Attribute Uniqueness Constraint Entries by Using Command-Line 19-11
19.6.7 Enabling and Disabling Attribute Uniqueness by Using Command-Line 19-12

20 Managing Knowledge References and Referrals

20.1 Introduction to Managing Knowledge References and Referrals 20-1
20.2 About Referral Sets 20-2
20.3 Configuring Smart Referrals 20-3
20.4 Configuring Default Referrals 20-4

21 Managing Directory Schema

21.1 Introduction to Managing Directory Schema 21-1
21.1.1 Understanding Directory Schema Management 21-1
21.1.2 Storage Location of Schema Information in the Directory 21-2
21.1.3 Understanding Object Classes 21-2
21.1.3.1 About Object Classes 21-3
21.1.3.2 Add Object Classes 21-4
21.1.3.3 Types of Modification to an Existing Object Class 21-4
21.1.3.4 Limitations for Deleting Object Classes 21-5
21.1.4 Understanding Attributes 21-5
21.1.4.1 Introduction to Attributes 21-5
21.1.4.2 Rules for Adding Attributes 21-6
21.1.4.3 Rules for Modifying Attributes 21-6
21.1.4.4 Rules for Deleting Attributes 21-6
21.1.4.5 Index option in Oracle Internet Directory to Search Attributes 21-7
21.1.5 Methods to Extend the Number of Attributes Associated with Entries 21-8
21.1.5.1 Extending the Number of Attributes Associated with Entries 21-9
21.1.5.2 Extending the Number of Attributes before Creating Entries in the Directory 21-9
21.1.5.3 Specifications to Extend the Number of Attributes for Existing Entries by Creating an Auxiliary Object Class 21-9
21.1.5.4 Specifications to Extend the Number of Attributes for Existing Entries by Creating a Content Rule 21-10
21.1.5.5 Rules for Creating and Modifying Content Rules 21-10
21.1.5.6 Schema Enforcement When Using Content Rules 21-11
21.1.5.7 Searches for Object Classes Listed in Content Rules 21-11
21.1.6 Understanding Attribute Aliases 21-12
21.1.7 Object Identifier Support in LDAP Operations 21-13
21.2 Managing Directory Schema by Using Oracle Directory Services Manager 21-13
21.2.1 Searching for Object Classes by Using Oracle Directory Services Manager 21-13
21.2.2 Adding Object Classes by Using Oracle Directory Services Manager 21-14
21.2.3 Modifying Object Classes by Using Oracle Directory Services Manager 21-14
21.2.4 Deleting Object Classes by Using Oracle Directory Services Manager 21-15
21.2.5 Viewing Properties of Object Classes by Using Oracle Directory Services Manager 21-16
21.2.6 Adding a New Attribute by Using Oracle Directory Services Manager 21-16
21.2.7 Modifying an Attribute by Using Oracle Directory Services Manager 21-17
21.2.8 Deleting an Attribute by Using Oracle Directory Services Manager 21-17
21.2.9 Viewing All Directory Attributes by Using Oracle Directory Services Manager 21-17
21.2.10 Searching for Attributes by Using Oracle Directory Services Manager 21-18
21.2.11 Adding an Index to a New Attribute by Using Oracle Directory Services Manager 21-18

21.2.12 Adding an Index to an Existing Attribute by Using Oracle Directory Services Manager 21-18
21.2.13 Dropping an Index from an Attribute by Using Oracle Directory Services Manager 21-19
21.2.14 Creating a Content Rule by Using Oracle Directory Services Manager 21-19
21.2.15 Modifying a Content Rule by Using Oracle Directory Services Manager 21-19
21.2.16 Viewing Matching Rules by Using Oracle Directory Services Manager 21-20
21.2.17 Viewing Syntaxes by Using Oracle Directory Services Manager 21-20
21.3 Managing Directory Schema by Using the Command Line 21-21
21.3.1 Viewing the Schema by Using ldapsearch 21-21
21.3.2 Adding a New Object Class by Using Command-Line Tools 21-21
21.3.3 Adding a New Attribute to an Auxiliary or User-Defined Object Class by Using Command-Line Tools 21-22
21.3.4 Modifying Object Classes by Using Command-Line Tools 21-22
21.3.5 Adding and Modifying Attributes by Using ldapmodify 21-23
21.3.6 Deleting Attributes by Using ldapmodify 21-23
21.3.7 Indexing an Attribute by Using ldapmodify 21-24
21.3.8 Dropping an Index from an Attribute by Using ldapmodify 21-24
21.3.9 Indexing an Attribute by Using the Catalog Management Tool 21-25
21.3.10 Adding a New Attribute With Attribute Aliases by Using the Command Line 21-25
21.3.11 Adding or Modifying Attribute Aliases in Existing Attributes by Using the Command Line 21-26
21.3.12 Deleting Attribute Aliases by Using the Command Line 21-26
21.3.13 Using Attribute Aliases with LDAP Commands 21-26
21.3.13.1 Using Attribute Aliases with ldapsearch 21-27
21.3.13.2 Using Attribute Aliases with ldapadd 21-28
21.3.13.3 Using Attribute Aliases with ldapmodify 21-28
21.3.13.4 Using Attribute Aliases with ldapdelete 21-29
21.3.13.5 Using Attribute Aliases with ldapmoddn 21-29
21.3.14 Managing Content Rules by Using Command-Line Tools 21-29
21.3.15 Viewing Matching Rules by Using ldapsearch 21-30
21.3.16 Viewing Syntaxes by Using ldapsearch 21-31

22 Configuring Referential Integrity

22.1 Introduction to Configuring Referential Integrity 22-1
22.2 Enabling Referential Integrity Using Fusion Middleware Control 22-2
22.3 Disabling Referential Integrity Using Fusion Middleware Control 22-2
22.4 Enabling Referential Integrity Using the Command Line 22-2
22.5 Configuring Specific Attributes for Referential Integrity by Using the Command Line 22-3
22.6 Disabling Referential Integrity by Using the Command Line 22-3
22.7 Detecting and Correcting Referential Integrity Violations 22-4

23 Managing Auditing

23.1 Introduction to Auditing 23-1
23.1.1 Configuring the Audit Store 23-2
23.1.2 Oracle Internet Directory Audit Configuration 23-2
23.1.3 Replication and Oracle Directory Integration Platform Audit Configuration 23-3
23.1.4 Audit Record Fields 23-3
23.1.5 Audit Record Storage 23-4
23.1.6 Generating Audit Reports 23-4
23.2 Managing Auditing Using Fusion Middleware Control 23-4
23.2.1 Auditing Oracle Internet Directory Sensitive Data Attributes 23-5
23.3 Managing Auditing Using WLST 23-6
23.4 Managing Auditing from the Command Line 23-7
23.4.1 Viewing Audit Configuration from the Command Line 23-7
23.4.2 Configuring Oracle Internet Directory Auditing from the Command Line 23-7
23.4.3 Enabling Replication and Oracle Directory Integration Platform Auditing 23-8

24 Managing Logging

24.1 Introduction to Logging 24-1
24.1.1 Oracle Internet Directory File Locations 24-1
24.1.2 Features of Oracle Internet Directory Debug Logging 24-2
24.1.3 Understanding Log Messages 24-3
24.1.3.1 Log Messages for Specified LDAP Operations 24-3
24.1.3.2 Log Messages Not Associated with Specified LDAP Operations 24-3
24.1.3.3 Example for Trace Messages in Oracle Internet Directory Server Log File 24-4
24.2 Managing Logging Using Fusion Middleware Control 24-5
24.2.1 Viewing Log Files Using Fusion Middleware Control 24-5
24.2.2 Configuring Logging Using Fusion Middleware Control 24-6
24.3 Managing Logging from the Command Line 24-6
24.3.1 Viewing Log Files from the Command Line 24-7
24.3.2 Setting Debug Logging Levels Using the Command Line 24-7
24.3.3 Setting the Debug Operation Using the Command Line 24-8
24.3.4 Force Flushing the Trace Information to a Log File 24-9

25 Monitoring Oracle Internet Directory

25.1 Introduction to Monitoring Oracle Internet Directory Server 25-1
25.1.1 Capabilities of Oracle Internet Directory Server Manageability 25-1
25.1.2 Oracle Internet Directory Server Manageability Architecture and Components 25-2
25.1.3 Security Events and Statistics Entries Purge 25-4
25.1.4 Account Used for Accessing Server Manageability Information 25-4

25.2 Overview of Statistics Collection Using Fusion Middleware Control 25-5
25.2.1 Configuring Directory Server Statistics Collection Using Fusion Middleware Control 25-5
25.2.2 Configuring a User for Statistics Collection Using Fusion Middleware Control 25-6
25.3 Overview of Statistics Information Viewable from Fusion Middleware Control 25-6
25.3.1 Statistics Information Viewable from the Oracle Internet Directory Home Page 25-7
25.3.2 Viewing Information on the Oracle Internet Directory Performance Page 25-7
25.4 Overview of Statistics Collection Using Metrics REST API 25-8
25.4.1 Enabling Oracle Internet Directory Metrics REST API 25-8
25.4.2 Securing Oracle Internet Directory Metrics REST API 25-8
25.4.3 Statistics Information Accessible from Metrics REST API 25-9
25.5 Statistics Information Accessible from the Oracle Directory Services Manager Home Page 25-9
25.6 Understanding Statistics Collection Using the Command-Line 25-9
25.6.1 Configuring Health, General, and Performance Statistics Attributes 25-10
25.6.2 Configuring Security Events Tracking 25-10
25.6.3 Configuring User Statistics Collection from the Command Line 25-11
25.6.4 Configuring Event Levels from the Command Line 25-11
25.6.5 Configuring a User for Statistics Collection Using the Command Line 25-12
25.7 Viewing Information with the OIDDIAG Tool 25-12
25.8 Monitoring Oracle Internet Directory Server Using LDAP 25-14
25.8.1 Viewing Monitoring Information Using the cn=monitor Entry 25-14
25.8.2 Viewing the Available Monitoring Information 25-14
25.8.3 Monitoring System Information 25-15
25.8.4 Monitoring the Work Queue 25-15
25.8.5 Monitoring Client Connections 25-15
25.8.6 Monitoring Version Information 25-16

26 Backing Up and Restoring Oracle Internet Directory

27 Managing Quality of Service Configuration

27.1 Understanding Quality of Service in Oracle Internet Directory 27-1
27.2 Configuring QoS to Restrict the Number of User Operations 27-1
27.3 Restricting the Number of Connections from one DN 27-2
27.4 Restricting the Number of Operations per Minute for a DN 27-2
27.5 Restricting the Number of Add Operations for a User 27-3
27.6 Restricting All Operations Using allop 27-3
27.7 Restricting the Number of Operations on a DSA Config Entry 27-3

Part III Advanced Administration: Security

28 Configuring Secure Sockets Layer (SSL)

28.1 Overview of Configuring Secure Sockets Layer (SSL) 28-1
28.1.1 Supported Cipher Suites 28-2
28.1.2 Supported Protocol Versions 28-4
28.1.3 About SSL Authentication Modes 28-5
28.1.4 SSL Authentication Modes 28-6
28.1.5 Oracle Wallets 28-7
28.1.6 Other Components and SSL 28-8
28.1.7 SSL Interoperability Mode 28-8
28.1.8 StartTLS 28-9
28.2 Overview of Configuring SSL by Using Fusion Middleware Control 28-9
28.2.1 Configuring SSL by Using Fusion Middleware Control 28-9
28.2.2 Creating a Wallet by Using Fusion Middleware Control 28-9
28.2.3 Configuring SSL Parameters by Using Fusion Middleware Control 28-11
28.2.4 SSL Parameters with Fusion Middleware Control 28-12
28.3 Overview of Configuring SSL by Using LDAP Commands 28-13
28.3.1 Configuring SSL Parameters by Using LDAP Commands 28-13
28.3.2 SSL Attributes 28-14
28.4 Configuring SSL in Oracle Internet Directory 28-14
28.5 Configuring ODSM Connection with SSL Enabled 28-15
28.6 Testing SSL Connections by Using Oracle Directory Services Manager 28-16
28.7 Overview of Testing SSL Connections From the Command Line 28-16
28.7.1 Testing SSL With Encryption Only 28-17
28.7.2 Testing SSL With Server Authentication 28-17
28.7.3 Testing SSL With Client and Server Authentication 28-17
28.8 Configuring SSL between Database and Oracle Internet Directory 28-18
28.8.1 Stopping an Instance of Oracle Internet Directory 28-18
28.8.2 Stopping Node Manager 28-18
28.8.3 Stopping Administration Server 28-18
28.8.4 Modifying the sqlnet.ora and listener.ora Files on the Database Server 28-18
28.8.5 Modifying the tnsnames.ora and sqlnet.ora configuration files on the OID Server 28-19
28.8.6 Setting the JAVA_OPTIONS Environment Variable on the Administration Server 28-19
28.8.7 Setting the JAVA_OPTIONS Environment Variable on the Node Manager 28-20
28.8.8 Restarting an Instance of Oracle Internet Directory 28-21

29 Configuring Data Privacy

29.1 Introduction to Table Space Encryption 29-1
29.2 Enabling and Disabling Table Space Encryption 29-1
29.2.1 Configuring First-time Settings for Table Space Encryption 29-2
29.2.2 Enabling or Disabling Table Space Encryption 29-3
29.3 Introduction to Using Database Vault With Oracle Internet Directory 29-3
29.4 Configuring Oracle Database Vault to Protect Oracle Internet Directory Data 29-4
29.4.1 Registering Oracle Database Vault with Oracle Internet Directory for First Time 29-4
29.4.2 Knowing Whether Oracle Database Vault is Registered with Oracle Database 29-4
29.4.3 Installing Bug Patches for Existing Oracle Database Vault Registration 29-5
29.4.4 Adding Database Vault Realm to Apply Policies 29-5
29.4.5 Enabling SQL*Plus Access to the Oracle Internet Directory Database 29-5
29.4.6 Blocking SQL*Plus Access to the Oracle Internet Directory Database 29-6
29.4.7 Database Vault Rules Defined for Oracle Internet Directory 29-6
29.4.8 Deleting Database Vault Policies For Oracle Internet Directory 29-6
29.4.9 Disabling Oracle Database Vault for the Oracle Internet Directory Database 29-6
29.5 Best Practices for Using Database Vault with Oracle Internet Directory 29-7
29.6 Introduction to Sensitive Attributes 29-7
29.6.1 List of Sensitive Attributes 29-7
29.6.2 Encryption Algorithm for Sensitive Attributes 29-8
29.7 Enabling Privacy Mode of Sensitive Attributes 29-8
29.8 Knowing Privacy Mode Status of Sensitive Attributes 29-9
29.9 Introduction to Hashed Attributes 29-9
29.10 Configuring Hashed Attributes 29-10
29.10.1 Configuring Hashed Attributes by Using Fusion Middleware Control 29-10
29.10.2 Configuring Hashed Attributes by Using ldapmodify 29-10

30 Managing Password Policies

30.1 Overview of Managing Password Policies 30-1
30.1.1 Introduction to Password Policy Rules 30-1
30.1.2 Creating and Applying a Password Policy 30-2
30.1.3 About Fine-Grained Password Policies 30-2
30.1.4 About Default Password Policy 30-4
30.1.5 Attributes for Password Policy 30-5
30.1.6 Operational Attributes of User Entry 30-7
30.1.7 About Directory Server Verification of Password Policy Information 30-7
30.1.8 About Password Policy Error Messages 30-8
30.2 Managing Password Policies by Using Oracle Directory Services Manager 30-9
30.2.1 Viewing Password Policies by Using Oracle Directory Services Manager 30-9
30.2.2 Modifying Password Policies by Using Oracle Directory Services Manager 30-9

30.2.3 Creating a Password Policy and Assigning it to a Subtree by Using ODSM 30-10
30.3 Managing Password Policies by Using Command-Line Tools 30-10
30.3.1 Viewing Password Policies by Using Command-Line Tools 30-10
30.3.2 Creating a New Password Policy by Using Command-Line Tools 30-11
30.3.3 Applying a Password Policy to a Subtree by Using Command-Line Tools 30-11
30.3.4 Setting Password Policies by Using Command-Line Tools 30-11
30.3.5 Making a Password Policy Entry Specific by Using Command-Line Tools 30-12
30.3.6 Determining Expired Users in Oracle Internet Directory by Using Command-Line Tools 30-13

31 Managing Directory Access Control

31.1 Overview of Directory Access Control 31-1
31.1.1 About Access Control Information 31-1
31.1.2 Introduction to Access Control Features 31-2
31.1.3 About Access Control Management Constructs 31-2
31.1.3.1 About Access Control Policy Points (ACPs) 31-3
31.1.3.2 About orclACI Attribute for Prescriptive Access Control 31-3
31.1.3.3 About orclEntryLevelACI Attribute for Entry-Level Access Control 31-3
31.1.3.4 About Security Groups 31-4
31.1.4 About Access Control Information Components 31-7
31.1.4.1 Access Control to Objects 31-8
31.1.4.2 Access Control to an Entity 31-9
31.1.4.3 Access Control to Operations 31-11
31.1.5 Access Level Requirements for LDAP Operations 31-12
31.1.6 ACL Evaluation 31-13
31.1.6.1 Attribute States During ACL Evaluation 31-13
31.1.6.2 Introduction to Precedence Rules Used in ACL Evaluation 31-14
31.1.6.3 Usage of More Than One ACI for the Same Object 31-15
31.1.6.4 Exclusionary Access to Directory Objects 31-16
31.1.6.5 About ACL Evaluation For Groups 31-16
31.2 Managing Access Control by Using Oracle Directory Services Manager 31-16
31.2.1 Viewing an ACP by Using Oracle Directory Services Manager 31-17
31.2.2 Adding an ACP by Using Oracle Directory Services Manager 31-17
31.2.2.1 Specifying the Entry That Will Be the ACP 31-17
31.2.2.2 Configuring Structural Access Items 31-18
31.2.2.3 Configuring Content Access Items 31-19
31.2.2.4 Deleting a Structural or Content Access Item 31-21
31.2.3 Modifying an ACP by Using Access Control Management in ODSM 31-21
31.2.4 Adding or Modifying an ACP by Using the Data Browser in ODSM 31-22
31.2.5 Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM 31-23
31.3 Managing Access Control by Using Command-Line Tools 31-23

31.3.1 Restricting the Kind of Entry a User Can Add 31-24
31.3.2 Setting Up an Inheritable ACP by Using ldapmodify 31-24
31.3.3 Setting Up Entry-Level ACIs by Using ldapmodify 31-25
31.3.4 Using Wildcards in an LDIF File with ldapmodify 31-25
31.3.5 Selecting Entries by DN 31-26
31.3.6 Using Attribute and Subject Selectors 31-26
31.3.7 Granting Read-Only Access 31-27
31.3.8 Granting Selfwrite Access to Group Entries 31-27
31.3.9 Defining a Completely Autonomous Policy to Inhibit Overriding Policies 31-27

32 Managing Password Verifiers

32.1 Password Verifiers for Directory Authentication 32-1
32.1.1 About Oracle Internet Directory 32-1
32.1.2 About Default Password Policy for Oracle Internet Directory 32-2
32.1.3 About Userpassword Verifiers and Directory Authentication 32-2
32.1.4 About Hashing Schemes for Creating Userpassword Verifiers 32-3
32.2 Managing Hashing Schemes for Password Verifiers for Directory Authentication 32-3
32.3 Password Verifiers for Components Authentication 32-3
32.3.1 About Password Verifiers for Oracle Components Authentication 32-4
32.3.2 Attributes for Storing Password Verifiers for Oracle Components Authentication 32-5
32.3.3 Default Verifiers for Oracle Components 32-7
32.3.4 Understanding Password Verification for an Oracle Component 32-8
32.4 Managing Password Verifier Profiles for Oracle Components by Using ODSM 32-9
32.5 Managing Password Verifier Profiles for Components by Using Command-Line Tools 32-9
32.5.1 Viewing a Password Verifier Profile by Using Command-Line Tools 32-9
32.5.2 Modifying a Password Verifier Profile by Using Command-Line Tools 32-10
32.6 Introduction to Verifiers Generation by Using Dynamic Parameters 32-10
32.7 Configuring Oracle Internet Directory to Generate Dynamic Password Verifiers 32-10

33 Delegating Privileges for Oracle Identity Management

33.1 Oracle Identity Management Privileges 33-1
33.1.1 About Privileges Delegation 33-1
33.1.2 About Delegation in an Oracle Fusion Middleware Environment 33-2
33.1.3 Default Configuration 33-3
33.1.4 Privileges for Administering the Oracle Technology Stack 33-3
33.2 User and Group Management Privileges 33-4
33.2.1 About Privileges for Managing User and Group Data 33-4
33.2.2 Default Privileges for Managing User Data 33-5
33.2.2.1 Privileges for Creating Users for a Realm 33-5
33.2.2.2 Privileges for Modifying Attributes of a User 33-6

33.2.2.3 Privileges for Deleting a User 33-6
33.2.2.4 Privileges for Delegating User Administration 33-6
33.2.3 Default Privileges for Managing Group Data 33-7
33.2.3.1 Privileges for Creating Groups 33-7
33.2.3.2 Privileges for Modifying the Attributes of Groups 33-8
33.2.3.3 Privileges for Deleting Groups 33-8
33.2.3.4 Privileges for Delegating Group Administration 33-8
33.3 Privileges for Deployment of Oracle Components 33-9
33.3.1 Granting Deployment Privileges 33-9
33.3.2 Oracle Application Server Administrators 33-10
33.3.3 User Management Application Administrators 33-10
33.3.4 Trusted Application Administrators 33-11
33.4 Delegating Privileges for Component Run Time 33-11
33.4.1 Authenticating Oracle Single Sign-On Server 33-12
33.4.2 Default Privileges for Reading and Modifying User Passwords 33-12
33.4.3 Default Privileges for Comparing User Passwords 33-13
33.4.4 Default Privileges for Comparing Password Verifiers 33-13
33.4.5 Default Privileges for Proxying on Behalf of End Users 33-14
33.4.6 Default Privileges for Managing the Oracle Context 33-14
33.4.7 Default Privileges for Reading Common User Attributes 33-15
33.4.8 Default Privileges for Reading Common Group Attributes 33-15
33.4.9 Default Privileges for Reading the Service Registry 33-15
33.4.10 Default Privileges for Administering the Service Registry 33-16

34 Managing Authentication

34.1 Introduction to Authentication 34-1
34.1.1 About Direct Authentication 34-1
34.1.2 About Indirect Authentication 34-3
34.1.3 About External Authentication 34-5
34.1.4 Simple Authentication and Security Layer (SASL) 34-5
34.1.4.1 Authenticating a SASL-Enabled Client to a Directory Server by Using Digest-MD5 34-5
34.1.4.2 Authenticating a SASL-Enabled Client to a Directory Server by Using External Authentication 34-6
34.2 About Certificate Authentication Method by Using Fusion Middleware Control 34-6
34.3 Configuring SASL Authentication by Using Oracle Enterprise Manager Fusion Middleware Control 34-6
34.4 Configuring Certificate Authentication Method by Using Command-Line Tools 34-7
34.5 SASL Authentication by Using the Command Line 34-8
34.6 Anonymous Binds 34-9
34.7 Managing Anonymous Binds 34-9
34.7.1 Managing Anonymous Binds by Using Fusion Middleware Control 34-9

34.7.2 Managing Anonymous Binds by Using the Command Line 34-10
34.8 Restricting Users from Binding to Oracle Internet Directory Server 34-10
34.9 Managing Unauthenticated Binds 34-11
34.9.1 Configuring Unauthenticated Binds 34-11
34.9.2 Managing Unauthenticated Binds by Using the Command Line 34-11

Part IV Advanced Administration: Managing Directory Deployment

35 Planning, Deploying and Managing Realms

35.1 Understanding Identity Management Realms 35-1
35.1.1 Overview of Identity Management Realm Planning 35-1
35.1.2 Understanding Identity Management Realms in an Enterprise Deployment 35-3
35.1.2.1 About Single Identity Management Realm in an Enterprise 35-3
35.1.2.2 About Multiple Identity Management Realms in an Enterprise 35-4
35.1.3 Overview of Identity Management Realms in a Hosted Deployment 35-4
35.1.4 Identity Management Realm Objects 35-5
35.1.5 Overview of Default Identity Management Realm 35-6
35.2 Customizing the Default Identity Management Realm 35-8
35.2.1 Default Identity Management Realm Customization 35-8
35.2.2 Understanding the Use Cases for Default Identity Management Realm Customization 35-9
35.2.3 Updating the Existing User and Group Search Base 35-10
35.2.4 Setting Up an Additional Search Base 35-11
35.2.5 Refreshing the Oracle Single Sign-On 35-12
35.2.6 Reconfiguring the Provisioning Profiles 35-13
35.3 About Additional Identity Management Realms for Hosted Deployments 35-14
35.4 Creating a DIT View 35-14
35.4.1 Specifying a DIT View 35-15
35.4.2 DIT View Syntax 35-15
35.4.3 About DIT View Conditions 35-16
35.4.4 Examples of DIT View 35-16

36 Tuning and Sizing Oracle Internet Directory

37 Managing Garbage Collection

37.1 Understanding Garbage Collection Management 37-1
37.1.1 Understanding the Components of the Garbage Collection Framework 37-1
37.1.1.1 About Garbage Collection Plug-in 37-1

37.1.1.2 Understanding the Background Database Processes 37-2
37.1.2 How Oracle Internet Directory Garbage Collection Works 37-4
37.1.3 About Garbage Collector Entries and Statistics Collector Entry 37-5
37.1.4 Overview of Change Log Purging 37-5
37.2 Setting Oracle Database Time Zone for Garbage Collection 37-6
37.3 Modifying the Oracle Internet Directory Garbage Collectors 37-7
37.3.1 Modifying a Garbage Collector by Using Oracle Directory Services Manager 37-7
37.3.2 Modifying a Garbage Collector by Using Command-Line Tools 37-7
37.3.2.1 Modifying a Garbage Collector 37-8
37.3.2.2 Disabling a Garbage Collector Change Log 37-8
37.3.3 Modifying the Oracle Internet Directory Statistics Collector 37-8
37.4 Managing Oracle Internet Directory Garbage Collectors Logging 37-8
37.4.1 Enabling the Oracle Internet Directory Garbage Collectors Logging 37-9
37.4.2 Disabling the Oracle Internet Directory Garbage Collectors Logging 37-10
37.4.3 Monitoring the Oracle Internet Directory Garbage Collection Logging 37-10
37.5 Configuring Time-Based Change Log Purging 37-10

38 Migrating Data from Other Data Repositories

38.1 Understanding Data Migration from Other Data Repositories 38-1
38.2 Migrating Data from LDAP-Compliant Directories 38-1
38.2.1 Understanding Data Migration Tools 38-2
38.2.1.1 Bulk Loader 38-2
38.2.1.2 Directory Integration Assistant 38-2
38.2.1.3 Features Comparison Between Bulk Loader and Directory Integration Assistant 38-2
38.2.1.4 LDIF File 38-3
38.2.2 Migrating LDAP Data Using LDIF File and Bulk Loader 38-3
38.2.3 Migrating LDAP Data Using Directory Integration Assistant Directly 38-5
38.2.4 Migrating LDAP Data Using LDIF File and Directory Integration Assistant 38-5
38.2.5 Migrating LDAP Data Using Directory Integration Assistant, Bulk Loader, and LDIF Files 38-6
38.2.6 Migrating LDAP Data Using the Oracle Directory Integration Platform Server 38-7
38.3 Migrating User Data from Application-Specific Repositories 38-7
38.3.1 Enabling Data Migration from Application-Specific Repositories 38-8
38.3.2 Reconciling Data in Application Repository with Existing Data in a Directory 38-8
38.3.3 Managing Data Migration from Application-Specific Repositories 38-8
38.3.3.1 Creating an Intermediate Template File 38-8
38.3.3.2 Running the OID Migration Tool 38-11

39 Configuring Directory Server Chaining

39.1 Understanding Directory Server Chaining Configuration 39-1
39.1.1 About Oracle Internet Directory Server Chaining 39-1
39.1.2 Supported External Directory Servers 39-2
39.1.3 Integrating Oracle Products with Oracle Internet Directory Server Chaining 39-2
39.1.3.1 Oracle Single Sign-On10g (10.1.4.3.0) or Later 39-2
39.1.3.2 Enterprise User Security 10g Only 39-2
39.1.4 Supported Operations for Server Chaining 39-3
39.1.5 About the Role of Server Chaining in Replication 39-3
39.2 Configuring Server Chaining 39-4
39.2.1 About Server Chaining Entries 39-4
39.2.2 Configuring Server Chaining by Using Oracle Directory Services Manager 39-4
39.2.3 Configuring Server Chaining from the Command Line 39-5
39.3 Creating Server Chaining Configuration Entries 39-6
39.3.1 Server Chaining Configuration Entry Attributes 39-7
39.3.2 Naming Conventions for User and Group Containers 39-8
39.3.3 Mapping of Oracle Internet Directory Attributes to External Directory Attributes 39-9
39.3.3.1 Default Attribute Mapping to Active Directory 39-9
39.3.3.2 Default Attribute Mapping to Sun Java System Directory Server 39-10
39.3.3.3 Default Attribute Mapping to Novell eDirectory 39-10
39.3.4 Example of Configuring an Active Directory for Server Chaining 39-10
39.3.5 Configuring an Active Directory for Server Chaining 39-11
39.3.6 Example of Configuring an Active Directory for Server Chaining with SSL 39-11
39.3.7 Configuring an Active Directory for Server Chaining with SSL 39-12
39.3.8 Adding New Attributes to an Existing Active Directory Server Chaining Entry 39-12
39.3.9 Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining 39-13
39.3.10 Configuring Sun Java System Directory Server (iPlanet) for Server Chaining 39-13
39.3.11 Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining with SSL 39-14
39.3.12 Configuring Oracle Directory Server Enterprise Edition and Sun Java System Directory Server (iPlanet) for Server Chaining with SSL 39-14
39.3.13 Example of Configuring an eDirectory for Server Chaining 39-15
39.3.14 Example of Configuring an eDirectory for Server Chaining with SSL 39-15
39.4 Debugging Server Chaining 39-16
39.5 Configuring an Active Directory Plug-in for Password Change Notification 39-16

40 Managing DIT Masking

40.1 About DIT Masking 40-1
40.2 DIT Masking Configuration Attributes 40-1
40.3 Configuring DIT Masking 40-2

40.3.1 Restricting Access by Container Name 40-2
40.3.2 Restricting Access by Entry Data 40-2
40.3.3 Disallowing Access to Containers from the Entire Directory 40-3

Part V Advanced Administration: Directory Replication

41 Setting Up Replication

41.1 Introduction to Setting Up Replication 41-1
41.1.1 Replication Transport Mechanisms 41-2
41.1.2 Replication Setup Methods 41-2
41.1.2.1 Command-line Tools to Setup and Modify Replication 41-2
41.1.2.2 Database Copy Procedure to Replicate from an Existing Host 41-3
41.1.3 Bootstrap Rules for Replication 41-3
41.1.4 The Replication Agreement 41-4
41.1.5 Other Replication Configuration Attributes 41-4
41.1.6 Replication Process and Architecture 41-4
41.1.7 Rules for Configuring LDAP-Based Replication 41-4
41.1.8 Replication Procedure for a Mixed Deployment of 10g and 11gR1 Nodes 41-6
41.1.9 Replication Security 41-6
41.1.9.1 Authentication of the Directory Replication Server 41-6
41.1.9.2 Use of SSL Encryption in Oracle Internet Directory Replication 41-7
41.1.10 LDAP Replication Filtering for Partial Replication 41-8
41.1.10.1 Filtering of Naming Contexts in LDAP Replication 41-8
41.1.10.2 Attributes that Control Naming Contexts 41-8
41.1.10.3 Filtering Rules for Naming Contexts in LDAP Replication 41-8
41.1.10.4 Scenarios of Filtering Naming Context in LDAP Replication 41-9
41.1.10.5 Rules for Including or Excluding Naming Contexts and Attributes 41-13
41.1.10.6 Optimization of Partial Replication Naming Context for Better Performance 41-14
41.2 Testing Replication by Using Oracle Directory Services Manager 41-16
41.3 Setting Up a LDAP-Based Replication by Using the Command Line 41-16
41.3.1 Copying Your LDAP Data by Using ldifwrite and bulkload 41-17
41.3.2 Setting Up an LDAP-Based Replica with Customized Settings 41-17
41.3.2.1 Data Migration Using ldifwrite/bulkload versus Automatic Bootstrapping 41-18
41.3.2.2 Setting Up an LDAP-Based Replica by Using Automatic Bootstrapping 41-18
41.3.2.3 Setting Up an LDAP-Based Replica by Using the ldifwrite Tool 41-22
41.3.3 Deleting an LDAP-Based Replica 41-26
41.3.3.1 Stopping the Directory Replication Server on the Node to be Deleted 41-27
41.3.3.2 Deleting the Replica from the Replication Group 41-27
41.4 Scenario: Setting Up a Multimaster Replication Group with Fan-Out 41-27
41.4.1 Understanding Multimaster Replication 41-28

41.4.2 Setting Up the Multimaster Replication Group for Node1 and Node2 41-29
41.4.3 Configuring the Replication Agreement 41-29
41.4.4 Starting the Replication Servers on Node1 and Node2 41-29
41.4.5 Testing the Directory Replication Between Node1 and Node2 41-29
41.4.6 Installing and Configuring Node3 as a Partial Replica of Node2 41-29
41.4.7 Customizing the Partial Replication Agreement 41-30
41.4.8 Starting the Replication Servers on All Nodes in the DRG 41-30
41.4.9 Installing and Configuring Node4 as a Full Replica of Node2 41-31
41.4.10 Testing the Replication from Node2 to Node4 41-31
41.4.11 Installing and Configuring Node5 as a Two-Way Replica of Node1 41-31
41.4.12 Testing the Two-Way Replication Between Node1 and Node5 41-31

42 Setting Up Replication Failover

42.1 Introduction to Replication Failover 42-1
42.1.1 Replication Failover Scenario 42-1
42.1.2 Supported Failover Topology Types 42-3
42.1.3 Limitations and Warnings for Replication Failover 42-4
42.1.4 Types of Replication Failover 42-5
42.2 Performing a Stateless Replication Failover 42-5
42.2.1 Stopping all Directory Replication Server on Related Nodes 42-6
42.2.2 Breaking Old Replication Agreement and Setting up New Agreement 42-6
42.2.3 Saving Last Change Number 42-6
42.2.4 Comparing and Reconciling New Supplier and Consumer 42-6
42.2.5 Updating Last Applied Change Number of New Agreement 42-7
42.2.6 Cleaning Up Old Agreement on Old Supplier 42-8
42.2.7 Starting All Directory Replication Server on Related Nodes 42-8
42.3 Performing a Time-Based Replication Failover 42-8
42.3.1 Configuring Change Log Garbage Collection Object on New Supplier 42-9
42.3.2 Saving Last Change Number from New Supplier 42-9
42.3.3 Enabling Change Log Regeneration on New Supplier 42-9
42.3.4 Waiting for the Desired Time Period to Elapse 42-10
42.3.5 Stopping all Directory Replication Servers on Related Nodes 42-10
42.3.6 Breaking Old Replication Agreement and Setting Up New Agreement 42-10
42.3.7 Updating Last Applied Change Number of the New Agreement 42-10
42.3.8 Cleaning Up Old Agreement on Old Supplier 42-11
42.3.9 Starting All Directory Replication Servers on Related Nodes 42-11

43 Managing Replication Configuration Attributes

43.1 Understanding Replication Configuration Attributes 43-1
43.1.1 Replication Configuration Container 43-1

43.1.2 Understanding Replica Subentry 43-1
43.1.2.1 About Replica Subentry 43-2
43.1.2.2 Replica Subentry Attributes 43-2
43.1.2.3 Example of Replica Subentry 43-2
43.1.3 Understanding Replication Agreement Entry 43-3
43.1.3.1 About Replication Agreement Entry 43-3
43.1.3.2 Replication Agreement Entry Attributes 43-3
43.1.3.3 About LDAP-Based Replication Agreements 43-5
43.1.3.4 Example of Two-Way LDAP-Based Replication Agreements 43-6
43.1.4 Replication Naming Context Container Entry 43-7
43.1.5 Understanding Replication Naming Context Object Entry 43-7
43.1.5.1 About Replication Naming Context Object Entry 43-7
43.1.5.2 Replication Naming Context Entry Attributes 43-7
43.1.5.3 Example of Replication Naming Context Entry 43-8
43.1.6 Understanding Replication Configuration Set 43-9
43.1.6.1 About Replication Configuration Set 43-9
43.1.6.2 Replication Configuration Set Attributes 43-9
43.1.7 Examples of Replication Configuration Objects in a Directory 43-11
43.2 Managing Replication Configuration Attributes Using the Command Line 43-14

44 Managing and Monitoring Replication

44.1 Introduction to Managing and Monitoring Replication 44-1
44.1.1 Implications of LDAP-Based Partial Replication 44-2
44.1.2 About Managing Worker Threads 44-2
44.1.3 Change Logs in Directory Replication 44-3
44.1.4 Overview of Change Log Partitioning in Directory Replication 44-3
44.1.4.1 About Change Log Partitioning in Directory Replication 44-3
44.1.4.2 Change Log Partitioning Strategy 44-4
44.1.4.3 Configuring Change Log Partitioning 44-4
44.1.5 The Human Intervention Queue 44-4
44.1.5.1 About Managing the Queues 44-5
44.1.5.2 About Queue Statistics 44-5
44.1.5.3 The Number of Entries the Human Intervention Queue Tools Can Process 44-5
44.1.6 About Pilot Mode 44-5
44.1.7 Overview of Conflict Resolution in Oracle Replication 44-6
44.1.7.1 About Conflict Resolution in Oracle Replication 44-6
44.1.7.2 Levels at Which Replication Conflicts Occur 44-6
44.1.7.3 Resolving Replication Conflicts Automatically 44-7
44.1.7.4 How Automated Conflict Resolution Works 44-7
44.2 Managing and Monitoring Replication by Using ODSM 44-8

44.2.1 Managing Local Change Logs by Using Oracle Directory Services Manager 44-8
44.2.1.1 Viewing Local Change Logs Using ODSM 44-9
44.2.1.2 Properties of the Change Log Entry 44-9
44.3 Overview of Managing and Monitoring Replication Using the Command Line 44-9
44.3.1 Enabling and Disabling Change Log Generation Using the Command Line 44-10
44.3.2 Overview of Viewing Change Logs Using ldapsearch 44-10
44.3.2.1 Viewing Change Logs Using ldapsearch 44-10
44.3.2.2 Important Attributes in the Change Log 44-11
44.3.3 Overview of Configuring Attributes of the Replica Subentry Using ldapmodify 44-11
44.3.3.1 Configuring Attributes of the Replica Subentry Using ldapmodify 44-11
44.3.3.2 Replica Subentry Attributes 44-12
44.3.4 Specifying Pilot Mode for a Replica by Using remtool 44-12
44.3.5 Overview of Configuring Replication Agreement Attributes by Using ldapmodify 44-13
44.3.5.1 Replication Agreement Options 44-13
44.3.5.2 Configuring Replication Agreement Attributes Using ldapmodify 44-13
44.3.6 Overview of Modifying Replica Naming Context Object Parameters Using ldapmodify 44-14
44.3.6.1 Modifying Replica Naming Context Object Parameters Using ldapmodify 44-14
44.3.6.2 Adding a Naming Context Object for an LDAP-Based Replica 44-15
44.3.6.3 Deleting a Naming Context Object 44-15
44.3.6.4 Modifying the orclIncludedNamingContexts Attribute for a Replica Naming Context Object 44-16
44.3.6.5 Modifying the orclExcludedNamingContexts Attribute for a Replica Naming Context Object 44-16
44.3.6.6 Modifying the orclExcludedAttributes Attribute for a Replica Naming Context Object 44-17
44.3.7 Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify 44-17
44.3.7.1 About Configuring Attributes of the Replication Configuration Set Using ldapmodify 44-17
44.3.7.2 Replication Configuration Attributes and Debug Levels 44-18
44.3.8 Overview of Monitoring Conflict Resolution Messages Using the Command Line 44-19
44.3.8.1 Monitoring Conflict Resolution Messages Using the Command Line 44-19
44.3.8.2 Conflict Resolution Messages 44-20
44.3.9 Managing the Human Intervention Queue 44-21
44.3.10 Monitoring Replication Progress in a Directory Replication Group Using remtool -pthput 44-22
44.3.11 About Viewing Queue Statistics and Verifying Replication Using remtool 44-23
44.3.12 Managing the Number of Entries the Human Intervention Queue Tools Can Process 44-24
44.3.13 Configuring Replication Filtering Using the orclEntryExclusionFilter Attribute 44-24
44.4 Overview of Comparing and Reconciling Inconsistent Data Using oidcmprec 44-25
44.4.1 Comparing and Reconciling Inconsistent Data Using oidcmprec 44-26

44.4.2 Conflict Scenarios 44-26
44.4.3 Operations Supported by oidcmprec 44-27
44.4.4 Output from oidcmprec 44-28
44.4.5 How oidcmprec Works 44-28
44.4.6 Source and Destination Directories Setup 44-29
44.4.7 DIT for the oidcmprec Operation 44-29
44.4.8 Attributes Selection for the Operation 44-30
44.4.9 Control of Change Log Generation 44-30
44.4.10 oidcmprec Command-Line Arguments Specification in a Text or XML Parameter File 44-31
44.4.11 Directory Schema Inclusion in oidcmprec 44-32
44.4.12 Override of Predefined Conflict Resolution Rules 44-32
44.4.13 User-Defined Compare and Reconcile Operation 44-33
44.4.14 Known Limitations of the oidcmprec Tool 44-33

Part VI Advanced Administration: Directory Plug-ins

45 Configuring a Customized Password Policy Plug-In

45.1 Configuring a Customized Password Policy Plug-in 45-1
45.2 Managing a Customized Password Policy Plug-in 45-2
45.2.1 Loading and Registering the PL/SQL Program 45-2
45.2.2 Coding the Password Policy Plug-in 45-3
45.2.3 Debugging the Password Policy Plug-in 45-3
45.2.4 Sample PL/SQL Package pluginpkg.sql Contents 45-4

46 Developing Plug-ins for the Oracle Internet Directory Server

46.1 Overview of Oracle Internet Directory Server Plug-in Framework 46-1
46.1.1 Supported Languages for Server Plug-ins 46-2
46.1.2 Prerequisites to Develop Server Plug-ins 46-3
46.1.3 Benefits of Using Server Plug-ins 46-3
46.1.4 Guidelines for Designing Server Plug-ins 46-3
46.1.5 Using the Server Plug-in Framework 46-4
46.1.6 LDAP Operations Supported by Oracle Internet Directory 46-4
46.1.7 Understanding LDAP Timings Supported by Oracle Internet Directory 46-4
46.1.7.1 About Pre-Operation Server Plug-ins 46-5
46.1.7.2 About Post-Operation Server Plug-ins 46-5
46.1.7.3 About When-Operation Server Plug-ins 46-5
46.1.7.4 About When_Replace-Operation Server Plug-ins 46-6
46.1.8 Using Plug-ins in a Replication Environment 46-6
46.1.9 Modifying JVM Options for Server Plug-ins 46-6

46.2 Creating a Plug-in 46-7
46.3 Registering a Plug-in From the Command Line 46-7
46.3.1 Object Classes and Attributes to Create a Plug-in Configuration Entry 46-7
46.3.2 Adding a Plug-in Configuration Entry by Using Command-Line Tools 46-10
46.4 Managing Plug-ins by Using Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control 46-11
46.4.1 Creating a Plug-in by Using Oracle Directory Services Manager 46-11
46.4.2 Registering a Plug-in by Using Oracle Directory Services Manager 46-11
46.4.3 Editing a Plug-in by Using Oracle Directory Services Manager 46-12
46.4.4 Deleting a Plug-in by Using Oracle Directory Services Manager 46-12
46.4.5 Managing JVM Options by Using Oracle Enterprise Manager Fusion Middleware Control 46-12

47 Configuring a Customized External Authentication Plug-in

47.1 Overview of Customized External Authentication Plug-in 47-1
47.2 Installing, Configuring, and Enabling the External Authentication Plug-in 47-2
47.3 Debugging the External Authentication Plug-in 47-3
47.4 Creating the PL/SQL Package oidexaup.sql 47-4

A Appendixes

A.1 Differences Between 11g and 12c A-1
A.1.1 Overview of Instance Creation and Process Management A-2
A.1.1.1 Creating 11g Oracle Internet Directory Instance A-2
A.1.1.2 Creating 12c Oracle Internet Directory Instance A-4
A.1.1.3 Starting and Stopping 11g Replication Server A-5
A.1.1.4 About Monitoring and Reporting the Status of Oracle Internet Directory Processes to OPMN by 11g OIDMON A-5
A.1.2 About Assigning SSL and non-SSL Ports A-5
A.1.3 Changed Path Names in 12c Configuration and Log Files A-6
A.1.4 About Configuring Audit Framework Using Oracle Enterprise Manager Fusion Middleware Control A-7
A.1.5 Updated Server Chaining A-7
A.1.6 About Setting Up and Managing LDAP-Based Replication A-7
A.1.7 About Java Containers A-7
A.2 Managing Oracle Internet Directory Instances by Using OIDCTL A-8
A.2.1 About Managing Oracle Internet Directory by Using OIDCTL A-8
A.2.2 Creating and Starting an Oracle Internet Directory Server Instance by Using OIDCTL A-10
A.2.3 About Stopping an Oracle Internet Directory Server Instance by Using OIDCTL A-10
A.2.4 About Starting an Oracle Internet Directory Server Instance by Using OIDCTL A-11
A.2.5 Viewing Status Information by Using OIDCTL A-11

A.2.6 Deleting an Oracle Internet Directory Server Instance by Using OIDCTL A-11
A.3 How Replication Works A-12
A.3.1 Architecture of LDAP-Based Replication A-12
A.3.2 LDAP Replica States A-13
A.3.3 Managing an Entry Using Multimaster Replication Process A-16
A.3.3.1 How the Multimaster Replication Process Adds a New Entry to a Consumer A-16
A.3.3.2 How the Multimaster Replication Process Deletes an Entry A-17
A.3.3.3 How the Multimaster Replication Process Modifies an Entry A-17
A.3.3.4 How the Multimaster Replication Process Modifies a Relative Distinguished Name A-18
A.3.3.5 How the Multimaster Replication Process Modifies a Distinguished Name A-19
A.4 Java Server Plug-in Developer's Reference A-20
A.4.1 Advantages of Java Plug-ins A-20
A.4.2 Setting Up a Java Plug-in A-21
A.4.3 orclPluginName Value A-21
A.4.4 Overview of Java Plug-in API A-22
A.4.4.1 Communication Between the Server and Plug-in A-22
A.4.4.2 Java Plug-in Structure A-23
A.4.4.3 Overview of PluginDetail A-23
A.4.4.4 PluginResult A-31
A.4.4.5 ServerPlugin Interface Methods for LDAP Operations A-31
A.4.5 Java Plug-in Error and Exception Handling Examples A-32
A.4.5.1 Run-time Exception Example A-33
A.4.5.2 Run-time Error Example A-33
A.4.5.3 PluginException Example A-33
A.4.6 Java Plug-in Debugging and Logging A-34
A.4.7 Java Plug-in Examples A-34
A.4.7.1 Example 1: Password Validation Plug-in A-35
A.4.7.2 Example 2: External Authentication Plug-in for Active Directory A-36
A.5 PL/SQL Server Plug-in Developer's Reference A-38
A.5.1 Designing, Creating, and Using PL/SQL Server Plug-ins A-38
A.5.1.1 PL/SQLPlug-in Caveats A-39
A.5.1.2 Specifications for PL/SQL Plug-in Package Names and Procedures A-39
A.5.1.3 Compiling PL/SQLPlug-ins A-41
A.5.1.4 Managing PL/SQL Plug-ins A-42
A.5.1.5 Enabling and Disabling PL/SQL Plug-ins A-42
A.5.1.6 Exception Handling in a PL/SQL Plug-in A-43
A.5.1.7 PL/SQL Plug-in LDAP API A-44
A.5.1.8 PL/SQL Plug-in and Database Tools A-45
A.5.1.9 Ensuring Security in PL/SQL Plug-ins A-45
A.5.1.10 PL/SQL Plug-in Debugging A-45

A.5.1.11 Specifications for PL/SQL Plug-in LDAP API A-46
A.5.2 Using PL/SQL Plug-ins A-46
A.5.2.1 Logging all ldapsearch Commands through a Plug-in A-46
A.5.2.2 Synchronizing Two DITs through a Plug-in A-48
A.5.3 Performing Binary Operations by using PL/SQLPlug-ins A-50
A.5.3.1 Modifying an Entry in Another Directory by using ldapmodify Plug-in A-51
A.5.3.2 Propagating a Change to Another Directory by using ldapadd Plug-in A-53
A.5.3.3 Comparing Binary Attributes by using ldapcompare Plug-in A-55
A.5.4 Object Type Definitions in the LDAP API Plug-in A-58
A.5.5 Specifications for PL/SQL Plug-in Procedures A-59
A.6 The LDAP Filter Definition A-62
A.6.1 Status of The String Representation of LDAP Search Filters A-63
A.6.2 IESG Note on The String Representation of LDAP Search Filters A-63
A.6.3 The String Representation of LDAP Search Filters Abstract A-63
A.6.4 LDAP Search Filter Definition A-64
A.6.5 String Search Filter Definition A-65
A.6.6 Using String Search Filters A-66
A.6.6.1 Using Simple String Search Filters A-66
A.6.6.2 Using String Search Filters with Extensible Matching A-67
A.6.6.3 Using String Search Filters with Escaping Mechanism A-67
A.6.7 Security Considerations in The String Representation of LDAP Search Filters A-67
A.6.8 References for The String Representation of LDAP Search Filters A-68
A.6.9 Address of The String Representation of LDAP Search Filters Author A-68
A.6.10 Copyright Notice in The String Representation of LDAP Search Filters A-68
A.7 The Access Control Directive Format A-69
A.7.1 Schema for orclACI A-69
A.7.2 Schema for orclEntryLevelACI A-70
A.8 Globalization Support in the Directory A-70
A.8.1 About Character Sets and the Directory A-71
A.8.1.1 About Unicode A-71
A.8.1.2 Unicode Implementations A-71
A.8.1.3 UTF-8 Support in Oracle Databases A-72
A.8.2 Components of the NLS_LANG Parameter A-72
A.8.3 Setting NLS_LANG Parameter from the Command Line A-73
A.8.4 Limitation of using Non-AL32UTF8 Databases A-73
A.8.5 Using Globalization Support with LDIF Files A-73
A.8.5.1 Interpretation of LDIF file Containing Only ASCII Strings A-73
A.8.5.2 Interpreting LDIF file Containing UTF-8 Encoded Strings A-74
A.8.6 Using Globalization Support with Command-Line LDAP Tools A-75
A.8.6.1 Enabling Command-line Tools to Convert Other Character Set Input to UTF-8 A-75
A.8.6.2 Specifying the -E Argument When Using Each Tool A-76

A.8.6.3 Using -E Argument with Command-Line LDAP Tools A-76
A.8.7 Setting NLS_LANG in the Client Environment A-77
A.8.8 Using Globalization Support with Bulk Tools A-78
A.8.8.1 Using Globalization Support with bulkload A-78
A.8.8.2 Using Globalization Support with ldifwrite A-78
A.8.8.3 Using Globalization Support with bulkdelete A-79
A.8.8.4 Using Globalization Support with bulkmodify A-79
A.9 Setting up Access Controls for Creation and Search Bases for Users and Groups A-79
A.9.1 Setting up Access Controls for the User Search Base and the User Creation Base A-80
A.9.2 Setting up Access Controls for the Group Search Base and the Group Creation Base A-82
A.10 Searching the Directory for User Certificates A-83
A.10.1 Mapping the Certificate A-83
A.10.1.1 Adding a Certificate Mapping Rule A-83
A.10.1.2 Deleting a Certificate Mapping Rule A-84
A.10.1.3 Modifying a Certificate Mapping Rule A-84
A.10.2 Search Types A-84
A.11 Adding a Directory Node by Using the Database Copy Procedure A-85
A.11.1 Definition of Sponsor Site and New Site in Database Copy Procedure A-86
A.11.2 Prerequisites for Database Copy Procedure A-86
A.11.3 Sponsor Directory Site Environment for Database Copy Procedure A-86
A.11.4 New Directory Site Environment for Database Copy Procedure A-87
A.11.5 Adding a Directory Node by using Database Copy Procedure A-87
A.11.5.1 Setting Up Sponsor Node A-87
A.11.5.2 Setting Up New Node A-92
A.11.5.3 Running LDAP-Based Replication A-97
A.12 RFCs Supported by Oracle Internet Directory A-98
A.13 Managing Oracle Directory Services Manager's Java Key Store A-100
A.13.1 Introduction to Managing ODSM's Java Key Store A-100
A.13.2 Retrieving ODSM's Java Key Store Password A-101
A.13.2.1 Retreiving Password Using Enterprise Manager Fusion Middleware Control A-101
A.13.2.2 Retrieving Password Using a Python Script A-103
A.13.3 Listing the Contents of odsm.cer Java Key Store A-104
A.13.4 Deleting Expired Certificates A-105
A.13.4.1 Determining the Expiration Date of a Certificate A-105
A.13.4.2 Deleting a Certificate A-106
A.14 Starting and Stopping the Oracle Stack A-106
A.14.1 Starting the Stack A-106
A.14.2 Stopping the Stack A-108
A.15 Performing a Rolling Upgrade A-108
A.15.1 About Rolling Upgrade A-109

A.15.2 Prerequisites for a Rolling Upgrade A-109
A.15.3 Performing a Rolling Upgrade A-109
A.15.4 Completing Post-Upgrade Task A-111
A.15.5 Rolling Upgrade Example A-111
A.16 Troubleshooting Oracle Internet Directory A-113
A.16.1 Problems and Solutions A-113
A.16.1.1 Installation Errors A-114
A.16.1.2 Oracle Database Server Errors A-114
A.16.1.3 Directory Server Error Messages and Causes A-117
A.16.1.4 Core Dump and Stack Trace Occurs When Oracle Internet Directory Crashes A-122
A.16.1.5 TCP/IP Problems A-122
A.16.1.6 Troubleshooting Password Policies A-123
A.16.1.7 Troubleshooting Directory Performance A-125
A.16.1.8 Troubleshooting Port Configuration A-128
A.16.1.9 Troubleshooting Starting Oracle Internet Directory A-128
A.16.1.10 Oracle Internet Directory Error Due to Interrupted Client Connection A-130
A.16.1.11 Troubleshooting Starting, Stopping, and Restarting of the Directory Server A-130
A.16.1.12 Troubleshooting Oracle Internet Directory Replication A-135
A.16.1.13 Troubleshooting Change Log Garbage Collection A-140
A.16.1.14 Troubleshooting Dynamic Password Verifiers A-141
A.16.1.15 Troubleshooting Oracle Internet Directory Password Wallets A-142
A.16.1.16 Troubleshooting bulkload Errors A-143
A.16.1.17 Troubleshooting bulkdelete, bulkmodify, and ldifwrite Errors A-144
A.16.1.18 Troubleshooting catalog Errors A-145
A.16.1.19 Troubleshooting remtool Errors A-145
A.16.1.20 Troubleshooting Server Chaining Error A-146
A.16.1.21 View Version Information A-146
A.16.1.22 Troubleshooting Oracle Enterprise Manager Fusion Middleware Control and WLST A-146
A.16.1.23 Troubleshooting Oracle Directory Services Manager A-147
A.16.1.24 Troubleshooting a Locked User Account A-150
A.16.1.25 Troubleshooting Policy Store Migration A-152
A.16.2 Need More Help? A-153
A.16.2.1 Oracle Internet Directory Debug Logs A-154

List of Figures

1-1	Oracle Internet Directory Overview	1-5
3-1	A Typical Oracle Internet Directory Node	3-3
3-2	Oracle Directory Server Instance Architecture	3-5
3-3	A Directory Information Tree	3-10
3-4	Attributes of the Entry for Anne Smith	3-12
3-5	Correct and Incorrect Naming Contexts	3-18
3-6	A Partitioned Directory	3-21
3-7	Using Knowledge References to Point to Naming Contexts	3-22
3-8	Placement of Resource Access and Resource Type Information in the DIT	3-29
4-1	Oracle Internet Directory Process Control Architecture	4-2
5-1	Planning the Directory Information Tree	5-2
6-1	A Replicated Directory	6-1
6-2	Example of Partial Replication	6-3
6-3	Example of Single-Master Replication	6-5
6-4	Example of Multimaster Replication	6-6
6-5	Example of Fan-Out Replication	6-7
6-6	Example of Multimaster Replication with Fan-Out	6-8
8-1	DIT Showing Two Instance-Specific Configuration Entries	8-2
8-2	Oracle Internet Directory Oracle Internet Directory Process Control Architecture	8-3
18-1	Alias Entries Example	18-2
18-2	Resulting Tree when Creating the My_file.ldif	18-3
19-1	Example of a Directory Information Tree	19-5
21-1	Location of Schema Components in Entries of Type subSchemaSubentry	21-2
25-1	Architecture of Oracle Internet Directory Server Manageability	25-3
30-1	Location of Password Policy Entries	30-3
30-2	pwdPolicy subentry Attributes Populated with DN of Password Policy	30-3
32-1	Location of the Password Verifier Profile Entry	32-4
32-2	Authentication Model	32-6
32-3	How Password Verification Works	32-8
33-1	Delegation Flow in an Oracle Fusion Middleware Environment	33-2
34-1	Indirect Authentication	34-3
35-1	Example of an Identity Management Realm	35-3
35-2	Enterprise Use Case: Single Identity Management Realm	35-4
35-3	Enterprise Use Case: Multiple Identity Management Realms	35-4
35-4	Hosted Deployment Use Case	35-5
35-5	Default Identity Management Realm	35-6

37-1	Example: Garbage Collection of Change Log Entries	37-4
37-2	Garbage Collection Entries in the DIT	37-5
38-1	Using an LDIF File and Bulk Loader	38-3
38-2	Using syncProfileBootstrap Directly	38-5
38-3	Using an LDIF File and syncProfileBootstrap	38-5
38-4	Using syncProfileBootstrap, bulkload, and LDIF Files	38-6
38-5	Using the Oracle Directory Integration Server	38-7
38-6	Structure of the Intermediate User File	38-9
41-1	A Sample Naming Context	41-9
41-2	Naming Context Object #1	41-10
41-3	Naming Context Object #2	41-11
41-4	Result of Combining Naming Context Objects #1 and #2	41-11
41-5	Naming Context Object #3	41-12
41-6	Naming Context Object #4	41-12
41-7	Result of Combining Naming Context Objects #3 and #4	41-13
41-8	Naming Context Object #5	41-14
41-9	Naming Context Object #6	41-15
41-10	Naming Context Object #7	41-16
41-11	Example of Fan-Out Replication	41-28
42-1	Replication Failover Scenario	42-2
42-2	Consumer and New Supplier Connected to Old Supplier by LDAP	42-3
42-3	Old and New Suppliers in the Same Directory Replication Group	42-4
42-4	Failover Preserving Replica Type	42-4
42-5	Compare and Reconcile All Connected Replicas	42-5
43-1	Example: Multimaster Replication and Fan-Out Replication	43-12
43-2	Example: Replication Configuration Entries for Node C	43-13
43-3	Example: Replication Configuration Entries for Node D	43-14
46-1	Oracle Internet Directory Plug-in Framework	46-2
A-1	A Component with Two Instances	A-9
A-2	LDAP Replication Process	A-12
A-3	Communication Between the Server and the Java Plug-in	A-22
A-4	getPortableCredential Operation	A-102
A-5	getPortableCredential Operation	A-103

List of Tables

1-1	Comparison of Online Directories and Relational Databases	1-2
3-1	An Oracle Internet Directory Node	3-3
3-2	Attributes Created with Each New Entry	3-13
3-3	Common LDAP Attributes	3-13
4-1	Process Control Items in the ODS_PROCESS_STATUS Table	4-3
6-1	Full or Partial Replication	6-3
6-2	Direction of Replication	6-3
6-3	Transport Protocols	6-4
6-4	Types of Directory Replication Groups	6-5
7-1	Using the Oracle Internet Directory Menu	7-6
7-2	Basic Tasks for Configuring and Managing Oracle internet Directory	7-17
9-1	Attributes of the Instance-Specific Configuration Entry	9-3
9-2	Attributes in the DSA Configuration Entry	9-10
9-3	Attributes of the DSE	9-14
9-4	Configuration Attributes on Server Properties Page, General Tab.	9-16
9-5	Configuration Attributes on Server Properties Page, Performance Tab	9-16
9-6	Configuration Attributes on Shared Properties Page, General Tab	9-18
9-7	Oracle Internet Directory-Related MBeans	9-21
14-1	orclDynamicGroup Attributes for "Connect By" Assertions	14-6
14-2	Static and Dynamic Group Considerations	14-11
17-1	Syntax Elements Used in Rules for Computed Attributes	17-2
18-1	Flags for Searching the Directory with Alias Entries	18-4
18-2	Entry Alias Dereferencing Messages	18-7
19-1	Attribute Uniqueness Constraint Entry	19-2
21-1	Attribute Aliases Used in Examples	21-26
23-1	Oracle Internet Directory Audit Configuration Attributes	23-2
23-2	Audit Configuration Attributes in Fusion Middleware Control	23-5
24-1	Oracle Internet Directory Log File Locations	24-1
24-2	Configuration Attributes on Server Properties Page, Logging Tab	24-6
24-3	Values for OrclDebugFlag	24-7
24-4	Debug Operations for Setting the orcldebugop Attribute	24-9
25-1	Components of Oracle Internet Directory Server Manageability	25-4
25-2	Configuration Attributes on Server Properties Page, Statistics Tab	25-5
25-3	Values of orcloptracklevel	25-10
25-4	Metrics Recorded by Each orcloptracklevel Value	25-11
25-5	Event Levels	25-12

28-1	TLS Cipher Suites Supported in Oracle Internet Directory	28-2
28-2	TLS Cipher Suites Supported in Oracle Internet Directory for Legacy Products	28-3
28-3	Protocol Mapping	28-5
28-4	SSL Authentication Modes	28-7
28-5	SSL-Related Attributes in Fusion Middleware Control	28-12
28-6	SSL Attributes	28-14
29-1	Sensitive Attributes Stored in orclencryptedattributes	29-7
29-2	LDAP and Bulk Operations on Attributes in orclhashedattributes	29-9
30-1	Password Policy Attributes	30-5
30-2	Password Policy-Related Operational Attributes	30-7
31-1	Sample Security Groups	31-6
31-2	Types of Access	31-11
31-3	LDAP Operations and Access Needed to Perform Each One	31-12
31-4	Attribute States During ACL Evaluation	31-13
31-5	DNs Used in Example	31-28
32-1	Attributes for Storing Password Verifiers in User Entries	32-5
33-1	Default Privileges Granted to Everyone and to Each User	33-3
33-2	Privileges for Administering the Oracle Technology Stack	33-3
33-3	Characteristics of the Subscriber DAS Create User Group	33-5
33-4	Characteristics of the Subscriber DAS Edit User Group	33-6
33-5	Characteristics of the DAS Delete User Group	33-6
33-6	Characteristics of the User Privilege Assignment Group	33-7
33-7	Characteristics of the Group Creation Group	33-7
33-8	Characteristics of the Group Edit Group	33-8
33-9	Characteristics of the Group Delete Group	33-8
33-10	Characteristics of the Group Privilege Assignment Group	33-9
33-11	Characteristics of the Oracle Application Server Administrators Group	33-10
33-12	Characteristics of the User Management Application Administrators Group	33-10
33-13	Characteristics of the Trusted Application Administrators Group	33-11
33-14	Characteristics of the User Security Administrators Group	33-13
33-15	Characteristics of the Authentication Services Group	33-13
33-16	Characteristics of the Verifier Services Group	33-14
33-17	Characteristics of the User Proxy Privilege Group	33-14
33-18	Characteristics of the Oracle Context Administrators Group	33-14
33-19	Characteristics of the Common User Attributes Group	33-15
33-20	Characteristics of the Common Group Attributes Group	33-15
33-21	Characteristics of the Service Registry Viewers Group	33-15

33-22	Characteristics of the Common Group Attributes Group	33-16
34-1	Direct Authentication Options	34-2
34-2	Configuration Attributes on Server Properties, SASL Tab	34-7
34-3	SASL Authentication Attributes	34-8
34-4	SASL Authentication Modes	34-8
34-5	Orclanonymousbindsflag Value and Directory Server Behavior	34-9
35-1	Oracle Identity Management Objects	35-5
35-2	Customizing the Default Identity Management Realm	35-8
35-3	DIT View Syntax	35-15
38-1	Features of bulkload and syncProfileBootstrap	38-2
38-2	List of Tasks to Migrate Data to Oracle Internet Directory using LDIF File and Directory Integration Assistant	38-6
38-3	List of Tasks to Migrate Data Using Directory Integration Assistant, Bulk Loader and LDIF Files	38-7
38-4	Mandatory Attributes in a User Entry	38-10
39-1	Configuration Entry Attributes for Server Chaining	39-7
39-2	Default Attribute Mapping to Active Directory	39-9
39-3	Default Attribute Mapping to Sun Java System Directory Server	39-10
39-4	Default Attribute Mapping to Novell eDirectory	39-10
40-1	Masking Configuration Attributes	40-1
41-1	Data Migration Using ldifwrite/bulkload versus Automatic Bootstrapping	41-18
41-2	Nodes in Example of Partial Replication Deployment	41-28
43-1	Attributes of the Replica Subentry	43-2
43-2	Attributes of the Replication Agreement Entry	43-3
43-3	Attributes of the Replication Naming Context Entry	43-8
43-4	Replication Configuration Set Attributes	43-9
44-1	Types of Replication Conflict	44-7
44-2	Properties on the Change Log Page	44-9
44-3	Important Attributes in the Change Log	44-11
44-4	Replica Subentry Attributes	44-12
44-5	Replication Agreement Options	44-13
44-6	Replication Configuration Attributes	44-18
44-7	Replication Debug Levels	44-19
44-8	Conflict Resolution Messages	44-20
46-1	Plug-in Configuration Objects and Attributes	46-7
A-1	Some Path Names that Changed	A-6
A-2	LDAP Replica States	A-14
A-3	Plug-in Names and Corresponding Paths	A-22

A-4	The Meaning of the DN Information for Each LDAP Operation	A-24
A-5	Behavior of Operation Result Code	A-25
A-6	Subclasses of LdapOperation and Class-specific information.	A-26
A-7	Behavior of LdapEntry Information for Each Plug-in Timing	A-26
A-8	Behavior of the AttributeName for Each Plug-in Timing	A-27
A-9	Behavior of the Attribute Value for Each Plug-in Timing	A-27
A-10	Behavior of the Delete DN for Each Plug-in Timing	A-27
A-11	Behavior of New Parent DN Information for Each Plug-in Timing	A-28
A-12	Behavior of New Relative DN Information for Each Plug-in Timing	A-28
A-13	Behavior of Delete Old RDN Information for Each Plug-in Timing	A-28
A-14	Behavior of LdapModification Information for Each Plug-in Timing	A-29
A-15	Behavior of the Required Attributes for Each Plug-in Timing	A-29
A-16	Behavior of the Scope for Each Plug-in Timing	A-30
A-17	Behavior of the SearchResultSet for Each Plug-in Timing	A-30
A-18	Debug Levels for Java Plug-in Logging	A-34
A-19	Plug-in Module Interface	A-40
A-20	Operation-Based and Attribute-Based Plug-in Procedure Signatures	A-40
A-21	Valid Values for the plug-in Return Code	A-43
A-22	Program Control Handling when a Plug-in Exception Occurs	A-44
A-23	Program Control Handling when an LDAP Operation Fails	A-44
A-24	Unicode Implementations	A-71
A-25	Components of the NLS_LANG Parameter	A-72
A-26	Examples: Using the -E Argument with Command-Line Tools	A-76
A-27	Supported RFCs	A-98
A-28	Standard Error Messages	A-118
A-29	Additional Error Messages	A-120
A-30	Password Policy Violation Error Messages	A-123
A-31	Error Messages for Dynamic Password Verifiers	A-141

Preface

The *Administering Oracle Internet Directory* describes Oracle Internet Directory concepts and architecture, and step-by-step instructions for performing basic and advanced administrative tasks.

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This document is intended for anyone who performs administration tasks for the Oracle Internet Directory. You should be familiar with either the UNIX operating system or the Microsoft Windows operating system to understand the line-mode commands and examples. You can perform all of the tasks through the command line utilities, and you can perform most of the tasks through Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control, which are operating system-independent.

To use this document, you need some familiarity with the Lightweight Directory Access Protocol (LDAP).

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:

- Online help available through Oracle Directory Services Manager and Oracle Fusion Middleware
- The Oracle Fusion Middleware and Oracle Database documentation sets, especially:
  - *Installing and Configuring Oracle Internet Directory*
  - *Reference for Oracle Identity Management*

  - *Administering Oracle Directory Integration Platform*
  - *Application Developer's Guide for Oracle Identity Management*

For additional information, see:

- Chadwick, David. *Understanding X.500—The Directory.* Thomson Computer Press, 1996.
- Howes, Tim and Mark Smith. *LDAP: Programming Directory-enabled Applications with Lightweight Directory Access Protocol.* Macmillan Technical Publishing, 1997.
- Howes, Tim, Mark Smith and Gordon Good, *Understanding and Deploying LDAP Directory Services.* Macmillan Technical Publishing, 1999.
- Internet Assigned Numbers Authority home page, `http://www.iana.org` for information about object identifiers
- Internet Engineering Task Force (IETF) documentation available at: `http://www.ietf.org`, especially:
  - The LDAPEXT charter and LDAP drafts
  - The LDAP charter and drafts
  - RFC 2254, "The String Representation of LDAP Search Filters"
  - RFC 1823, "The LDAP Application Program Interface"
- The OpenLDAP Community, `http://www.openldap.org`

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.
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 Oracle Internet Directory?

This section provides a brief description of the new features introduced with the latest release of Oracle Internet Directory and points you to more information about each new feature.

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

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

Oracle Internet Directory 14c (14.1.2.1.0) has the following key new features:

- **JDK Upgrade:** Oracle Internet 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.
- **Deprecating TLS 1.1:** Support for TLS 1.1 and earlier versions has been discontinued. Oracle Internet Directory now requires TLS 1.2 or TLS 1.3 for secure communication.

Part I
Understanding Directory Services

Part I explains what Oracle Internet Directory is and some of the concepts you must know before using it. It contains these chapters:

- Introduction to Directory Services
- Understanding Oracle Internet Directory in Oracle Fusion Middleware
- Understanding the Concepts and Architecture of Oracle Internet Directory
- Understanding Process Control of Oracle Internet Directory Components
- Understanding Oracle Internet Directory Organization
- Understanding Oracle Internet Directory Replication

1

Introduction to Directory Services

The topics in this section introduces online directories, provides an overview of the Lightweight Directory Application Protocol (LDAP) version 3, and explains some of the unique features and benefits of Oracle Internet Directory.

- Understanding Directory and its Role
- What is Lightweight Directory Access Protocol (LDAP)?
- Understanding Oracle Internet Directory
- How Oracle Products Use Oracle Internet Directory

1.1 What is a Directory?

A directory is a hierarchically organized collection of entries with similar attributes. Directories list resources—for example, people, books in a library, or merchandise in a department store—and give details about each one. A directory can be either offline—for example, a telephone book or a department store catalog—or online.

Online directories are used by enterprises with distributed computer systems for fast searches, management of users and security, and integration of multiple applications and services. Online directories have become critical to e-businesses and hosted environments.

1.2 Understanding Directory and its Role

You can understand about the types of directories and its role from this section.

This section contains the following topics:

- The Expanding Role of Online Directories
- The Problem: Too Many Special-Purpose Directories

1.2.1 The Expanding Role of Online Directories

An online directory is a specialized database that stores and retrieves collections of information about objects. Such information can represent any resources that require management: employee names, titles, and security credentials; information about partners; or information about shared network resources such as conference rooms and printers.

Online directories can be used by a variety of users and applications, and for a variety of purposes, including:

- An employee searching for corporate white page information, and, through a mail client, looking up e-mail addresses
- An application, such as a message transport agent, locating a user's mail server
- A database application identifying role information for a user

Although an online directory is a database—that is, a structured collection of data—it is not a relational database. The following table contrasts online directories with relational databases.

Table 1-1 Comparison of Online Directories and Relational Databases

Online Directories	Relational Databases
Designed to handle relatively simple transactions on relatively small units of data. For example, an application might use a directory simply to store and retrieve an e-mail address, a telephone number, or a digital portrait.	**Designed to handle large and diverse transactions using many operations on large units of data.**
Designed to be location-independent. Directory-enabled applications expect, at all times, to see the same information throughout the deployment environment—regardless of which server they are querying. If a queried server does not store the information locally, then it must either retrieve the information or point the client application to it transparently.	**Typically designed to be location-specific.** While a relational database can be distributed, it usually resides on a particular database server.
Designed to store information in entries. These entries might represent any resource customers want to manage: employees, e-commerce partners, conference rooms, or shared network resources such as printers. Associated with each entry are several attributes, each of which may have one or more values assigned. For example, typical attributes for a `person` entry might include first and last names, e-mail addresses, the address of a preferred mail server, passwords or other login credentials, or a digitized portrait.	**Designed to store information as rows in relational tables.**

1.2.2 The Problem: Too Many Special-Purpose Directories

In the past, some large companies had more than a hundred different directories, each designated for a special purpose. In addition, some applications had their own additional directories of user names.

Managing so many special purpose directories caused several problems, including:

- High cost of administration: Administrators had to maintain essentially the same information in many different places. For example, when an enterprise hired a new employee, administrators created a new user identity on the network, created a new e-mail account, added the user to the human-resources database, and set up all applications that the employee might need—for example, user accounts on development, testing, and production database systems. Later, if the employee left the company, administrators had to reverse the process to disable all these user accounts.
- Inconsistent data: Because of the large administrative overhead, it was difficult for multiple administrators, entering redundant information in multiple systems, to synchronize this employee information across all systems. The result was inconsistent data across the enterprise.
- Security issues: Each separate directory had its own password policy—which means that a user had to learn a variety of user names and passwords, each for a different system.

Today's enterprises need a more general purpose directory infrastructure, one based on a common standard for supporting a wide variety of applications and services.

1.3 What is Lightweight Directory Access Protocol (LDAP)?

The topics in this section introduce you to LDAP, a standard, extensible directory access protocol that directory clients and servers use to communicate.

- LDAP and Simplified Directory Management
- LDAP Version 3

1.3.1 LDAP and Simplified Directory Management

LDAP was conceived as an Internet-ready, lightweight implementation of the International Standardization Organization (ISO) X.500 standard for directory services. It requires a minimal amount of networking software on the client side, which makes it particularly attractive for Internet-based, thin client applications.

The LDAP standard simplifies management of directory information in three ways:

- It provides all users and applications in the enterprise with a single, well-defined, standard interface to a single, extensible directory service. This makes it easier to rapidly develop and deploy directory-enabled applications.
- It reduces the need to enter and coordinate redundant information in multiple services scattered across the enterprise.
- Its well-defined protocol and array of programmatic interfaces make it more practical to deploy Internet-ready applications that leverage the directory.

1.3.2 LDAP Version 3

The most recent version of LDAP, Version 3, was approved as a proposed Internet Standard by the Internet Engineering Task Force (IETF) in December 1997.

LDAP Version 3 improves on LDAP Version 2 in several important areas:

- Globalization Support: LDAP Version 3 allows servers and clients to support characters used in every language in the world.
- Knowledge references (also called referrals): LDAP Version 3 implements a referral mechanism that allows servers to return references to other servers as a result of a directory query. This makes it possible to distribute directories globally by partitioning a directory information tree (DIT) across multiple LDAP servers.
- Security: LDAP Version 3 adds a standard mechanism for supporting Simple Authentication and Security Layer (SASL), providing a comprehensive and extensible framework for data security.
- Extensibility: LDAP Version 3 enables vendors to extend existing LDAP operations by using mechanisms called controls. These are extra pieces of information carried along with existing operations, altering the behavior of the operation. When a client application passes a control along with the standard LDAP command, the behavior of the commanded operation is altered accordingly. For example, when a client wants to modify meta-information hidden in the directory, it can send the manageDSAIT control along with the LDAP command.
- Feature and schema discovery: LDAP Version 3 enables publishing information useful to other LDAP servers and clients, such as the supported LDAP protocols and a description of the directory schema.

See Also:

- RFCs (Requests for Comments) 2251-2256 of the IETF, available at: `http://www.ietf.org`
- RFCs Supported by Oracle Internet Directory
- Related Documents for an additional list of resources on LDAP
- Understanding the Concepts and Architecture of Oracle Internet Directory for a conceptual discussion of directory information trees and knowledge references
- About LDAP Controls in *Reference for Oracle Identity Management* for a list and description of controls supported by Oracle Internet Directory

1.4 Understanding Oracle Internet Directory

You can understand about the overview, components and advantages of Oracle Internet Directory in the following sections.

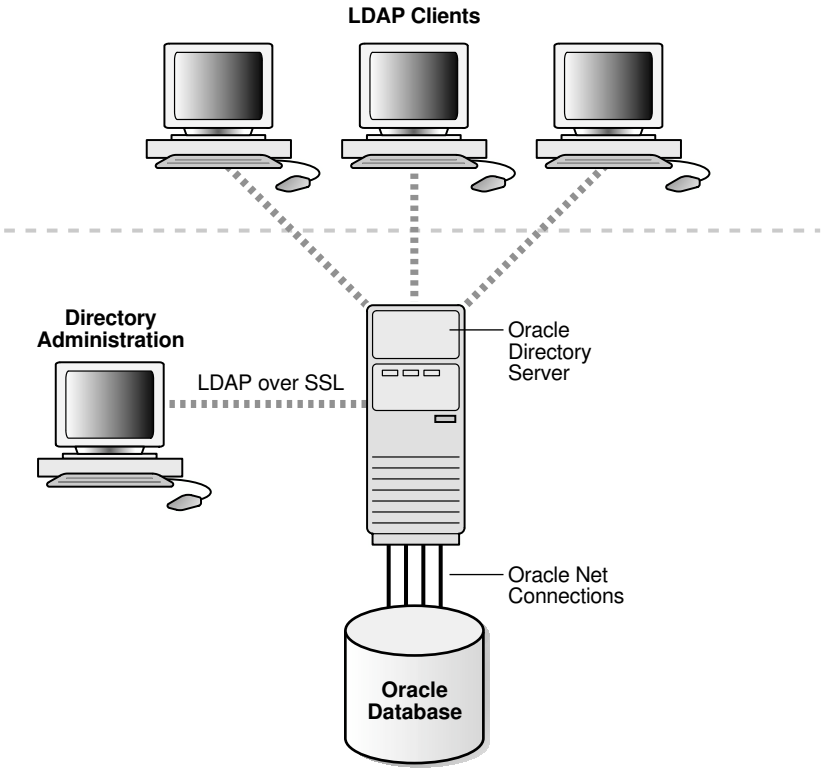
- Overview of Oracle Internet Directory
- Components of Oracle Internet Directory
- Advantages of Oracle Internet Directory

1.4.1 Overview of Oracle Internet Directory

Oracle Internet Directory is a general purpose directory service that enables fast retrieval and centralized management of information about dispersed users and network resources. It combines Lightweight Directory Access Protocol (LDAP) Version 3 with the high performance, scalability, robustness, and availability of an Oracle Database.

Oracle Internet Directory runs as an application on an Oracle Database. It communicates with the database by using Oracle Net Services, Oracle's operating system-independent database connectivity solution. The database may or may not be on the same host. Figure 1-1 illustrates this relationship.

Figure 1-1 Oracle Internet Directory Overview



1.4.2 Components of Oracle Internet Directory

You can understand about the components of Oracle Internet Directory in the following section.

Oracle Internet Directory includes:

- Oracle directory server, which responds to client requests for information about people and resources, and to updates of that information, by using a multitiered architecture directly over TCP/IP
- Oracle directory replication server, which replicates LDAP data between Oracle directory servers
- Directory administration tools, which include:
  - The Oracle Internet Directory pages in Oracle Enterprise Manager Fusion Middleware Control
  - Oracle Directory Services Manager
  - Command-line administration and data management tools
- Oracle Internet Directory Software Developer's Kit

> **See Also:**
>
> Developing Applications for Oracle Identity Management in *Application Developer's Guide for Oracle Identity Management* in the Oracle Internet Directory Software Developer's Kit

1.4.3 Advantages of Oracle Internet Directory

Among its significant benefits, Oracle Internet Directory provides scalability, high availability, security, and tight integration with the Oracle environment.

- Scalability in Oracle Internet Directory
- High Availability in Oracle Internet Directory
- Security Benefits in Oracle Internet Directory
- Integration of Oracle Internet Directory with Oracle Environment

1.4.3.1 Scalability in Oracle Internet Directory

Oracle Internet Directory exploits the strengths of an Oracle Database, enabling support for terabytes of directory information. In addition, such technologies as shared LDAP servers and database connection pooling enable it to support thousands of concurrent clients with subsecond search response times.

Oracle Internet Directory has a multi-threaded, multi-process and multi-instance physical architecture. This brings great flexibility to deployment options. Oracle Internet Directory can scale with a very high number of CPUs on SMP or NUMA hardware. With Oracle RAC database and the Oracle Internet Directory cluster configuration, you can deploy a single directory on multiple hardware nodes for horizontal scalability.

Oracle Internet Directory also provides data management tools, such as Oracle Directory Services Manager and a variety of command-line tools, for manipulating large volumes of LDAP data.

> **See Also:**
>
> High Availability Concepts in *Oracle Fusion Middleware High Availability Guide*

1.4.3.2 High Availability in Oracle Internet Directory

Oracle Internet Directory offers the most comprehensive high availability configurations. Directory replication, active/passive cluster configuration, active/active cluster configuration with Oracle RAC Database, and Disaster Recovery configurations with Oracle Data Guard.

Oracle Internet Directory also takes advantage of all the availability features of the Oracle Database. Because directory information is stored securely in the Oracle Database, it is protected by Oracle's backup capabilities. Additionally, the Oracle Database, running with large data stores and heavy loads, can recover from system failures quickly.

> **See Also:**
>
> High Availability Concepts in *Oracle Fusion Middleware High Availability Guide*

1.4.3.3 Security Benefits in Oracle Internet Directory

Oracle Internet Directory offers comprehensive and flexible access control. An administrator can grant or restrict access to a specific directory object or to an entire directory subtree. Moreover, Oracle Internet Directory implements three levels of user authentication: anonymous, password-based, and certificate-based using Secure Sockets Layer (SSL) Version 3 for authenticated access and data privacy.

With Oracle Database Vault, Oracle Internet Directory can restrict administrators and privileged users from accessing Directory data. With Oracle Transparent Data Encryption, Oracle Internet Directory can ensure protection of data on disk as well as on backups.

1.4.3.4 Integration of Oracle Internet Directory with Oracle Environment

Through Oracle Directory Integration Platform, Oracle Internet Directory provides a single point of integration between the Oracle environment and other directories such as NOS directories, third-party enterprise directories, and application-specific user repositories.

1.5 How Oracle Products Use Oracle Internet Directory

Oracle products use Oracle Internet Directory for easier administration, tighter security, and simpler integration between multiple directories.

- Easier and More Cost-Effective Administration of Oracle Products
- Tighter Security Through Centralized Security Policy Administration in Oracle Internet Directory
- Integration of Multiple Directories with Oracle Internet Directory

1.5.1 Easier and More Cost-Effective Administration of Oracle Products

Various Oracle products use Oracle Internet Directory. Some of them are described in the following section.

Oracle Platform Security Services stores users and groups in an embedded LDAP repository by default. Domains can be configured, however, to use identity data in LDAP repositories, such as Oracle Internet Directory. In addition, Oracle WebLogic Server provides a generic LDAP authenticator that can be used with other LDAP servers. By default, OPSS stores policies and credentials in file-based stores. These stores can be changed (or reassociated) to an LDAP repository backed by an Oracle Internet Directory or an Oracle Virtual Directory server.

Oracle Webcenter Suite bases its security on OPSS. It can delegate enforcement to Oracle Internet Directory for identity, policy, and credential storage. You can use LDAP commands to add or modify users and to search the directory, which can be useful when exporting and importing user accounts.

Oracle Access Manager supports storing user, configuration, and policy data in directories, such as Oracle Internet Directory (multiple realms). You can store data either together on the same directory server or on different directory servers.

Oracle Net Services uses Oracle Internet Directory to store and resolve database services and the simple names, called net service names, that can be used to represent them.

1.5.2 Tighter Security Through Centralized Security Policy Administration in Oracle Internet Directory

The **Oracle Database** uses Oracle Internet Directory to store user names and passwords, along with authorization information such as enterprise roles. It uses Oracle Internet Directory to store a password verifier along with the entry of each user.

Oracle Enterprise User Security uses Oracle Internet Directory for:

- Central Management of user authentication credentials

  Instead of storing a user's database password in each database, Oracle Advanced Security stores it in one place: the directory. It stores the password as an attribute of the user entry.

- Central management of user authorizations

  Oracle Advanced Security uses directory entries, called enterprise roles, to determine the privileges for a given enterprise user within a given schema, whether that schema is shared or owned. Enterprise roles are containers for database-specific global roles. For example, a user might be assigned the enterprise role of clerk, which might contain the global role of hr clerk with its attendant privileges on the human resources database and the global role of analyst with its attendant privileges on the payroll database.

- Mappings to shared schemas

  Oracle Advanced Security uses mappings—that is, directory entries that point an enterprise user to shared application schemas on the database instead of to an individual account. For example, you might map several enterprise users to the schema `sales_application` instead of to separate accounts in their names.

- Single password authentication

  In the Oracle Database, Oracle Advanced Security enables enterprise users to authenticate to multiple databases by using a single, centrally managed password. The password is stored in the directory as an attribute of the user's entry and is protected by encryption and access control lists. This spares you from setting up Secure Sockets Layer (SSL) on clients and users from having to remember multiple passwords.

- Central storage of PKI credentials

  In Oracle Database and Oracle Application Server, user wallets can be stored in the directory as an attribute of the user's entry. Storing wallets in this manner enables mobile users to retrieve and open their wallets by using Enterprise Login Assistant. While the wallet is open, authentication is transparent—that is, users can access any database on which they own or share a schema without having to authenticate again.

1.5.3 Integration of Multiple Directories with Oracle Internet Directory

Integrating multiple directories with Oracle Internet Directory is described in the following section.

Another directory services product, Oracle Virtual Directory, provides a single, dynamic access point to multiple data sources through LDAP or XML protocols. It does this by providing a real-time data join and an abstraction layer that exposes a single logical directory, without the need to synchronize or move data from its native location. Oracle Virtual Directory can provide multiple application-specific views of identity data stored in, for example, Oracle Internet Directory, Microsoft Active Directory and Sun Java Systems Directory instances, and can also be used to secure data access to the application-specific sources and enhance high-availability

to existing data-sources. These capabilities accelerate the deployment of applications and reduce costs by eliminating the need to consolidate user information before an application can be deployed. Oracle Virtual Directory can constantly adapt those applications to a changing identity landscape as user repositories are added, changed, or removed. Oracle Virtual Directory provides the following benefits:

- Consolidates multiple directories
- Provides virtualization and distribution of directory services
- Reduces administrative cost and improves security
- Extends enterprise applications quickly
- Provides ubiquitous access to information
- Lowers cost of implementation

Oracle Directory Integration Platform is a collection of interfaces and services for integrating multiple directories by using Oracle Internet Directory and several associated plug-ins and connectors. It provides these benefits:

- All Oracle components are pre-certified to work with Oracle Internet Directory.
- You can integrate the entire Oracle environment with third-party directories simply by integrating each third-party directory with Oracle Internet Directory. This saves you from having to integrate each application with each directory.

See Also:

Concepts and Architecture of Connected Directory Integration in *Administering Oracle Directory Integration Platform*

2

Understanding Oracle Internet Directory in Oracle Fusion Middleware

Features of Oracle Fusion Middleware that affect Oracle Internet Directory is described in the following sections. Since 11g Release 1 (11.1.1.0.0), Oracle Internet Directory was integrated with Oracle Fusion Middleware, a common management infrastructure that uses Oracle WebLogic Server.

- Understanding WebLogic Server Domain
- Oracle Internet Directory as a System Component
- Oracle Internet Directory Deployment Options
- Middleware Home
- WebLogic Server Home
- Oracle Home
- Oracle Instance in 14c Release
- Oracle Enterprise Manager Fusion Middleware Control
- Known Limitations of Oracle Enterprise Manager for Oracle Internet Directory
- Logging, Auditing and Diagnostics Using Fusion Middleware Control
- MBeans and the WebLogic Scripting Tool

> **See Also:**
>
> Introduction to Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

2.1 Understanding WebLogic Server Domain

A WebLogic Server administration domain is a logically related group of Java components. Domains include a special WebLogic Server instance called the Administration Server, which is the central point from which you configure and manage all resources in the domain.

Usually, you configure a domain to include additional WebLogic Server instances called managed servers. You deploy Java components, such as Web applications and Web services, and other resources onto the managed servers and use the Administration Server for configuration and management purposes only. The managed servers can be grouped together into a cluster.

2.2 Oracle Internet Directory as a System Component

Oracle Internet Directory is a system component. That is, it is a manageable process that is not an Oracle WebLogic Server. System components can use the WebLogic Administrative Domain for management services, including Oracle Enterprise Manager Fusion Middleware

Control, Audit Framework, configuration management through MBeans and Secure Sockets Layer and Wallet Management. The Oracle WebLogic Server Administration Server controls Oracle Internet Directory and other system components.

Oracle Internet Directory itself is a C-based process. Its only run time dependency is the Oracle Database. To be managed by the Oracle Fusion Middleware management framework, Oracle Internet Directory must register itself with a local or a remote Oracle WebLogic Server administration domain during installation or from the command line after installation. Therefore, an Oracle Internet Directory 14*c* installation requires either a local or a remote installation of Oracle WebLogic Server. Also, the Directory Management user interface, ODSM, is a Java component deployed on Oracle WebLogic Server.

If you must manage Oracle Internet Directory in your deployment using only command-line tools and a remote ODSM, there is also an option to install and configure Oracle Internet Directory without registering with a Oracle WebLogic Server Domain.

2.3 Oracle Internet Directory Deployment Options

During installation, you can choose deployment options for Oracle Internet Directory.

The four deployment options are:

1. **Create New Domain**–Oracle Internet Directory with a local Oracle WebLogic Server Domain. Oracle WebLogic Server is installed locally with Oracle Internet Directory and an admin domain is created for Oracle Internet Directory.
2. **Extend Existing Domain**–Oracle Internet Directory with a remote Oracle WebLogic Server Domain. Oracle WebLogic Server admin server and domain have been installed and created separately and Oracle Internet Directory registers with the Domain remotely.
3. **Expand Cluster**–Oracle Internet Directory in an Oracle WebLogic Server cluster for High Availability. This option will not be discussed here.
4. **Configure Without Domain**–Oracle Internet Directory without an Oracle WebLogic Server Domain. Oracle Internet Directory can be installed and configured without Oracle WebLogic Server Server and without registering to any Oracle WebLogic Server Admin Domains. In this case, Oracle Internet Directory cannot be managed by Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Scripting Tool (WLST), or other common Oracle Fusion Middleware management services. You must rely on command-line utilities such as `wlst` and the LDAP tools. ODSM can be deployed separately and used to manage Oracle Internet Directory.

If you choose **Create New Domain** or **Extend Existing Domain**, the Oracle Internet Directory component you create is registered with that domain when the installation is complete.

If you choose **Configure Without Domain**, the Oracle Internet Directory component is not registered with any domain when the installation is complete. You will be unable to manage Oracle Internet Directory, or any other component in that Oracle instance, with Oracle Enterprise Manager Fusion Middleware Control until you register the component with a WebLogic domain by using the command-line tool `wlst`.

2.4 Middleware Home

A Middleware home consists of the WebLogic Server home, and optionally one or more other Oracle product homes (also known as Oracle homes).

A middleware home can reside on a local file system or on a remote shared disk that is accessible through NFS. The Oracle Fusion Middleware home is represented in path names as *MW_HOME*.

2.5 WebLogic Server Home

A WebLogic Server home contains installed files necessary to host a WebLogic Server.

The WebLogic Server home directory is a peer of other Oracle home directories underneath the middleware home directory. In path names, it is represented as *WLS_HOME*.

2.6 Oracle Common Home

The Oracle home that contains the binary and library files required for the Oracle Enterprise Manager Fusion Middleware Control and Java Required Files (JRF).

There can be only one Oracle Common home within each Middleware home. In path names, it is represented as *ORACLE_COMMON_HOME*.

2.7 Oracle Home

An Oracle home contains installed files necessary to host a specific software suite. An Oracle home resides within the directory structure of the Middleware home. Each Oracle home can be associated with multiple Oracle instances.

The Oracle home is usually represented in path names as *ORACLE_HOME*. Each Oracle home can be associated with multiple Oracle instances or WebLogic server domains.

2.8 Oracle Instance in 14c Release

In 14c Release, product configuration data has been separated from product binaries. The product binaries reside in the Oracle home, *ORACLE_HOME*, while updatable files reside in an Oracle instance, represented in path names as *DOMAIN_HOME*.

Most Oracle Internet Directory commands require that you set the environment variable `DOMAIN_HOME` to the value of *DOMAIN_HOME*. You dereference this variable as `$DOMAIN_HOME` on UNIX or Linux systems and as `%DOMAIN_HOME%` on Windows.

All configuration files, repositories, log files, deployed applications, and temporary files reside in a oracle instance. Keeping updatable files separate from non-updatable files facilitates administrative tasks such as patching, upgrades, backup and restore, and cloning. It allows administrators to have their run-time and install-time binaries follow independent life cycles.

Domain refers not only to a physical location on disk but also encompasses the associated processes. The domain contains one or more active middleware system components, such as Oracle Virtual Directory or Oracle Internet Directory. You determine which components are part of an instance, either at install time or by creating and configuring an instance at a later time.

When you install Oracle Internet Directory on a host computer, Oracle Identity Management 14*c* Installer creates an Oracle Fusion Middleware component of type OID in a new or existing domain. The component name for the first Oracle Internet Directory component is `oid1`.

Oracle Identity Management 14*c* Installer also creates some directories in the file system under the domain, including the following, when you install Oracle Internet Directory:

```
$DOMAIN_HOME/config/fmwconfig/components/OID/config/componentName
$DOMAIN_HOME/servers/OID/logs/componentName
$DOMAIN_HOME/tools/OID/logs
$DOMAIN_HOME/config/fmwconfig/components/OID/admin
$DOMAIN_HOME/config/fmwconfig/components/OID/admin
$DOMAIN_HOME/tools/OID/load
```

2.9 Oracle Enterprise Manager Fusion Middleware Control

As of release 14*c*, you create, configure, and manage many Oracle Internet Directory features by using Oracle Enterprise Manager Fusion Middleware Control and Oracle Directory Services Manager.

Fusion Middleware Control enables you to configure and manage all Oracle products from one user interface. You can perform most configuration functions in Fusion Middleware Control that you can perform from the command line.

Oracle Directory Services Manager is an additional administrative interface for Oracle Internet Directory and Oracle Virtual Directory. It is accessible from Oracle Enterprise Manager Fusion Middleware Control or directly from its own URL.

2.10 Known Limitations of Oracle Enterprise Manager for Oracle Internet Directory

Oracle Enterprise Manager Fusion Middleware Control has the following limitations for Oracle Internet Directory in 14c:

- The wizard for setting up replication is no longer available in Oracle Enterprise Manager Fusion Middleware Control 14c Administration menu. You can use the command line tools for setting up LDAP-based replication. You can use the command line tools for setting up LDAP-based replication. See Command-line Tools to Setup and Modify Replication.
- The wizard for estimating sizing and tuning needs is no longer available in Oracle Enterprise Manager Fusion Middleware Control 14c Administration menu. For recommendations on sizing and tuning Oracle Internet Directory, see Tuning and Sizing Oracle Internet Directory.
- The wallet option is no longer available in Oracle Enterprise Manager Fusion Middleware Control 14c Security menu. You can use the orapki tool or the keystore service to create a wallet, see Wallet Management and Keystore Management in *Administering Oracle Fusion Middleware*.

2.11 Logging, Auditing and Diagnostics Using Fusion Middleware Control

Using Oracle Enterprise Manager Fusion Middleware Control, you can monitor the Oracle Internet Directory Server and related components and activities. Using the monitoring functions, you can gain insight into system activity and performance, for example, total logins, successful and unsuccessful logins, average login time, request latencies, LDAP connections, and so on.

You can monitor the following items:

- **Metrics**: To monitor system health

- **General**: A high-level rollup of load, performance, security, login, CPU utilization, and other data
- **Performance**: Key metrics for the directory server and its host
- **Reports**: Data on operation success and failure
- **Topology**: Information on the Oracle HTTP Server instances, directory server instances, associated databases, and other components

2.12 MBeans and the WebLogic Scripting Tool

The Oracle WebLogic Scripting Tool (WLST) is a command-line scripting environment that you can use to create, manage, and monitor Oracle WebLogic Server domains. It is based on the Java scripting interpreter, Jython. You can use WLST to perform some Oracle Internet Directory management operations.

A managed bean (MBean) is a Java object that represents a JMX manageable resource in a distributed environment, such as an application, a service, a component or a device. When Oracle Internet Directory is registered with an Oracle WebLogic Server Admin Domain, Oracle Internet Directory MBeans are deployed in the Oracle WebLogic Server Admin Server. These MBeans enable management of Oracle Internet Directory configuration through Oracle Enterprise Manager Fusion Middleware Control or WLST.

> **See Also:**
>
> Getting Started Using the Oracle WebLogic Scripting Tool (WLST) in *Administering Oracle Fusion Middleware*

3

Understanding the Concepts and Architecture of Oracle Internet Directory

Understand about the description of Oracle Internet Directory architecture and conceptual descriptions of Oracle Internet Directory basic elements such as directory entities, attributes, object classes, naming contexts, and security management.
This section contains the following topics:

- Understanding the Architecture of Oracle Internet Directory
- Understanding How Oracle Internet Directory Processes a Search Request
- Understanding Directory Entries in Oracle Internet Directory
- Understanding the Concept of Attributes in Oracle Internet Directory
- Understanding Object Classes in Oracle Internet Directory
- Directory Naming Contexts
- Security Features in Oracle Internet Directory
- Globalization Support
- Understanding Distributed Directories
- Knowledge References and Referrals
- Service Registry and Service to Service Authentication
- Oracle Directory Integration Platform
- Understanding the Role of Identity Management in Oracle Internet Directory
- TCP Keep-Alive Mechanism
- Understanding the Concept of Resource Information

> **See Also:**
>
> Related Documents for suggestions on further reading about LDAP-compliant directories

3.1 Understanding the Architecture of Oracle Internet Directory

Understand about the contextual description of the various components in Oracle Internet Directory.

This section contains the following topics:

- Oracle Internet Directory Node
- Oracle Directory Server Instance
- Oracle Internet Directory Ports

- Directory Metadata

3.1.1 Oracle Internet Directory Node

An Oracle Internet Directory node consists of one or more directory server instances connected to the same directory store. The directory store—the repository of the directory data—is an Oracle Database.

> **Note:**
>
> All Oracle Internet Directory instances in the same domain connect to the same Oracle Database.

Figure 3-1 shows the various directory server elements and their relationships running on a single node.

Oracle Net Services is used for all connections between the Oracle database server and:

- The Oracle directory server non-SSL port (3060 by default)
- The Oracle directory server SSL-enabled port (3131 by default)
- The OID Monitor

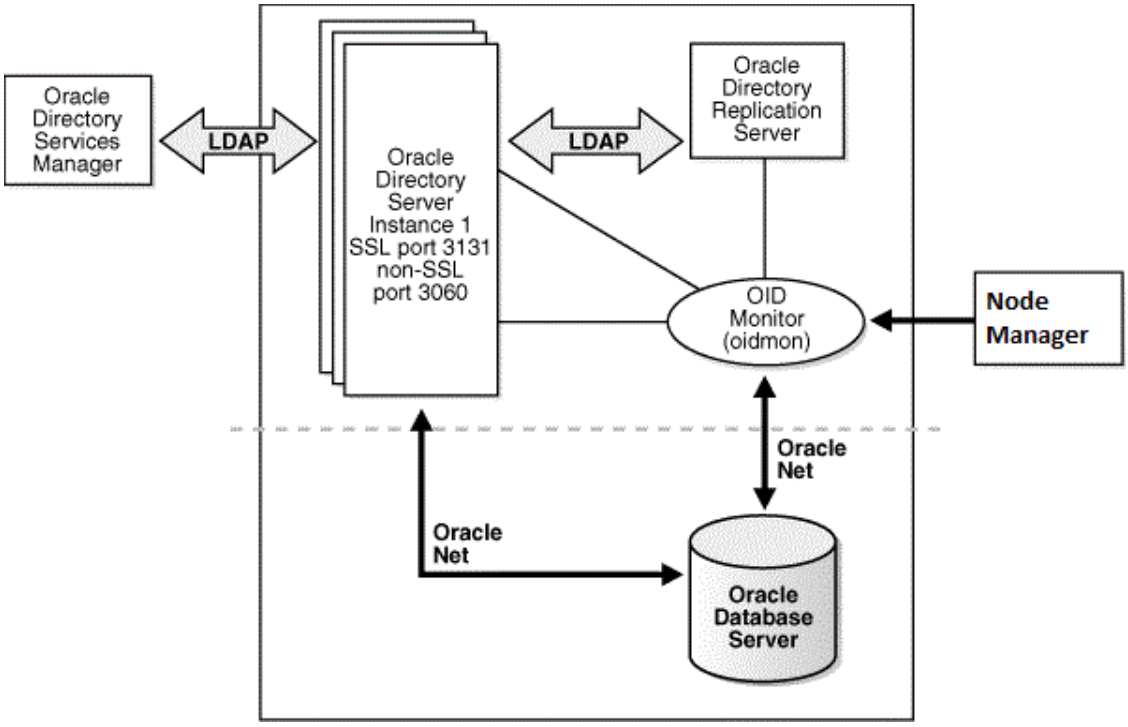
LDAP is used for connections between directory server and:

- Oracle Directory Services Manager
- Oracle directory replication server

The Oracle directory server instance and the Oracle directory replication server connect to OID Monitor by way of the operating system.

> **Note:**
>
> Beginning with Oracle Internet Directory 11*g* Release 1 (11.1.1.7.0), you can specify that Oracle Internet Directory server call OCIPing() to send keep alive messages to its Oracle Database. The frequency of these messages is determined by the new `orclMaxTcpIdleConnTime` attribute.
>
> Setting this attribute to a value less than the timeout value of the firewall between Oracle Internet Directory server and the Oracle Database prevents the Database connection from being dropped.
>
> For more information, see Attributes of the DSA Configuration Entry.

Figure 3-1 A Typical Oracle Internet Directory Node



As shown in Figure 3-1, an Oracle Internet Directory node includes the following major elements:

Table 3-1 An Oracle Internet Directory Node

Element	Description
Oracle directory server instance	Also called either an LDAP server instance or a directory server instance, it services directory requests through a single Oracle Internet Directory dispatcher process listening at specific TCP/IP ports. There can be more than one directory server instance on a node, listening on different ports.
Oracle directory replication server	Also called a replication server, it tracks and sends changes to replication servers in another Oracle Internet Directory system. There can be only one replication server on a node. You can choose whether to configure the replication server.
Oracle Database Server	Stores the directory data. The database can reside on the same node as the directory server instances. **Note:** To meet the need of Oracle Internet Directory's potentially large data volume as customer data size grows and to achieve continuous optimized database query performance, Oracle strongly recommends that you dedicate a database for exclusive use by the directory.

Table 3-1 (Cont.) An Oracle Internet Directory Node

Element	Description
OID Monitor (OIDMON)	Initiates, monitors, and terminates the LDAP server and replication server processes. When you invoke process management commands, such as `oidctl`, when you use Fusion Middleware Control to start or stop server instances, your commands are interpreted by this process. OIDMON also monitors servers and restarts them if they have stopped running for abnormal reasons. OIDMON starts a default instance of OIDLDAPD. If the default instance of OIDLDAPD is stopped using the OIDCTL command, then OIDMON stops the instance. However, when OIDMON is started by `wlst start()`, OIDMON restarts the default instance. All OID Monitor activity is logged in the file *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName`*`/oidmon-xxxx.log`. This file is on the Oracle Internet Directory server file system. OID Monitor checks the state of the servers through mechanisms provided by the operating system.
OID Control Utility (OIDCTL)	Communicates with OID Monitor by placing message data in Oracle Internet Directory server tables. This message data includes configuration parameters required to run each Oracle directory server instance. Normally used from the command line only to stop and start the replication server. OIDCTL is also used for checking the status of Oracle Internet Directory.

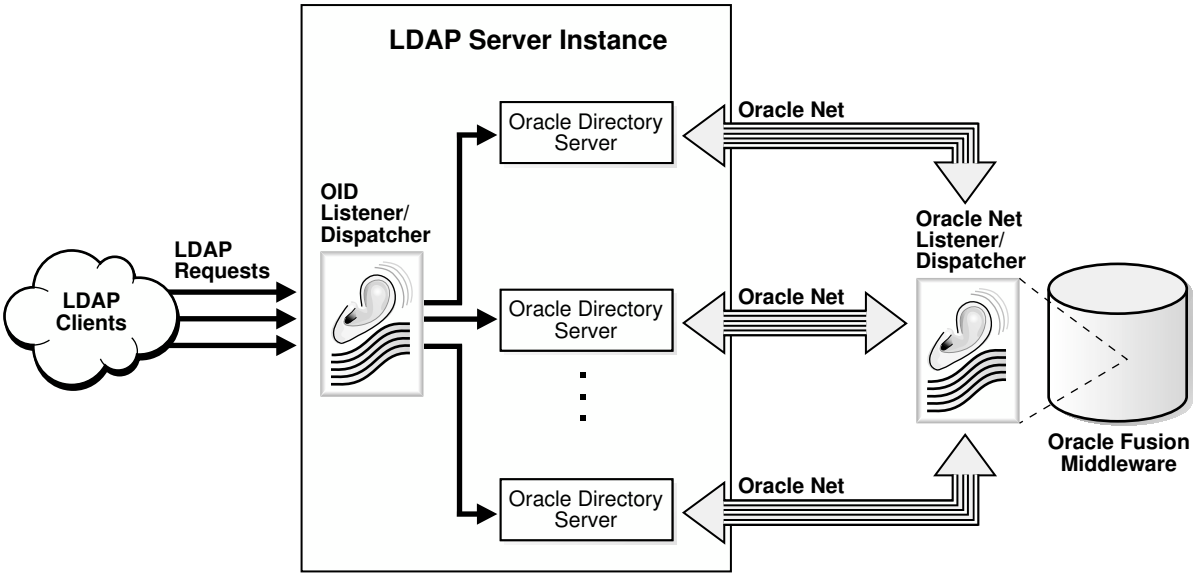
The Oracle directory replication server uses LDAP to communicate with an Oracle directory (LDAP) server instance. To communicate with the database, all components use OCI/Oracle Net Services. Oracle Directory Services Manager and the command-line tools communicate with the Oracle directory servers over LDAP.

3.1.2 Oracle Directory Server Instance

You can understand about Oracle Directory Server Instance architecture from the illustration and the description provided in this topic.

Figure 3-2 illustrates an Oracle directory server instance, also called an LDAP server instance.

Figure 3-2 Oracle Directory Server Instance Architecture



One instance comprises one dispatcher process and one or more server processes. The Oracle Internet Directory dispatcher and server processes can use multiple threads to distribute the load. LDAP clients send LDAP requests to an Oracle Internet Directory listener/dispatcher process listening for LDAP commands at its port.

Oracle Internet Directory listener/dispatcher starts a configured number server process at startup time. The number of server processes is controlled by the `orclserverprocs` attribute in the instance-specific configuration entry. The default value for `orclserverprocs` is 1. Multiple server processes enable Oracle Internet Directory to take advantage of multiple processor systems.

The Oracle Internet Directory dispatcher process sends the LDAP connections to the Oracle Internet Directory server process in a round robin fashion. The maximum number of LDAP connections accepted by each server is 1024 by default. This number can be increased by changing the attribute `orclmaxldapconns` in the instance-specific configuration entry, which has a DN of the form:

`cn=`*`componentname`*`,cn=osdldapd,cn=subconfigsubentry`

3.1.3 Oracle Internet Directory Ports

An Oracle Internet Directory component can be created by Oracle Identity Management 14*c* Installer or by a command-line tool. The program that creates Oracle Internet Directory follows specific steps in assigning the SSL and non-SSL port.

First, it attempts to use 3060 as the non-SSL port. If that port is unavailable, it tries ports in the range 3061 to 3070, then 13060 to 13070.

Similarly, the program that creates Oracle Internet Directory attempts to use 3131 as its SSL port, then ports in the range 3132 to 3141, then 13131 to 13141.

3.1.4 Directory Metadata

Directory metadata is the information used by the directory server during run time for processing LDAP requests. It is stored in the underlying data repository. During startup, the

directory server reads this information and stores it in a local metadata cache. It then uses this cache during its run time to process incoming LDAP operation requests.

> **Note:**
>
> The metadata cache is a write-through cache. An LDAP operation first writes to the database and then invalidates the corresponding cache entry. A subsequent search of that entry causes the cache to be refreshed.

The directory server has the following types of metadata in its local metadata cache:

- Directory Schema

  The definitions of object classes, attributes, and matching rules supported by the directory server. The directory server uses this information during creation and modification of directory objects. A directory object is a collection of object classes and their associated attributes and matching rules. See Managing Directory Schema .

- Access control policy point (ACP)

  A directory administrative domain for defining and controlling access to the information in that domain. The directory server uses ACPs when determining whether to allow a certain LDAP operation performed by a user. See Managing Directory Access Control.

- Root DSE entry

  The root DSE (Directory Service Agent-Specific Entry) contains several attributes that store information about the directory server itself. For example, these attributes contain the following information items, to mention just a few:

  – Naming contexts DNs
  – Sub Schema Subentry DN
  – Superior references (referrals) DNs
  – Special entry DNs like Oracle Internet Directory configuration and registry containers
  – Special Entry DNs like change log and change status containers
  – DN of replications agreement container

  See "Attributes of the DSE".

- Privilege groups

  Groups that can be used in access control policies.

  The directory schema supports directory group objects through the standard `groupofuniquenames` and `groupofnames` object classes. These object classes hold information for such groups as distribution lists and mailing lists to mention just two.

  Oracle Internet Directory extends these standard group objects through an auxiliary object class called `orclprivilegegroup`. This object class, which supports privilege groups that can be used in access control policies, provides flexibility to grant or deny access to groups of users. The directory server uses this information during:

  – LDAP bind operations to find out the subscribed privileged groups for a given user
  – Access control policy evaluation if the policy has directives that grant or deny access to privileged groups

> **See Also:**
>
> Managing Dynamic and Static Groups in Oracle Internet Directory

- Catalog entry

  A special entry containing information about indexed attributes in the underlying database. The directory uses this information during directory search operations. See "Index option in Oracle Internet Directory to Search Attributes ".

- Common entry

  A special entry containing information about hosted companies. A hosted company is an enterprise to which another enterprise provides services. The metadata in this entry includes the hosted company DN, user search base, nickname and other attributes, all of which are described in Planning, Deploying and Managing Realms.

- Plug-in entry

  A special entry containing information about the kind of operation that triggers a plug-in event, and the point in the operation when that plug-in is to be triggered. Developing Plug-ins for the Oracle Internet Directory Server, describes this information.

- Password verifier entry

  A special entry containing information about the encryption and verifier attribute types. Managing Password Verifiers , describes this information.

- Password policy entry

  One or more special entries containing information about policies enforced by the directory server for the user password credentials. The directory server uses this information during run time to enforce the password policies. See Managing Password Policies .

3.2 Understanding How Oracle Internet Directory Processes a Search Request

Understand about how Oracle Internet Directory processes a standard and persistent LDAP search operations.

The following topics provide a contextual description of the various LDAP search operations in Oracle Internet Directory:

- Standard LDAP Search Operations
- Persistent LDAP Search Operations

3.2.1 Standard LDAP Search Operations

Follow the procedure to process standard LDAP search request.

This example shows you how Oracle Internet Directory processes a search request.

1. The user or client enters a search request that is conditioned by one or more of the following options:

   - SSL: The client and server can establish a session that uses SSL encryption and authentication, or SSL encryption only. If SSL is not used, the client's message is sent in clear text.

- Type of user: The user can seek access to the directory either as a particular user or as an anonymous user, depending on which of the two has the necessary privileges to perform the desired function.
- Filters: The user can narrow the search by using one or more search filters, including those that use the Boolean conditions "and," "or," and "not," and those that use other operators such as "greater than, "equal to," and "less than."

2. The C API, using the LDAP protocol, sends a request to a directory server instance to connect to the directory.
3. The directory server authenticates the user, a process called binding. The directory server also checks the Access Control Lists (ACLs) to verify that the user is authorized to perform the requested search.
4. The directory server converts the search request from LDAP to Oracle Call Interface (OCI)/ Oracle Net Services and sends it to the Oracle Database.
5. The Oracle Database retrieves the information and passes it back through the chain—to the directory server, then to the C API, and, finally, to the client.

> **Note:**
>
> The maximum number of attributes you can specify in a search filter is 255.

3.2.2 Persistent LDAP Search Operations

A persistent search continues after the initial search results are returned by the Oracle Internet Directory server to the LDAP client. After the initial search is finished, the connection to the server is kept alive until the client unbinds or abandons the operation. A persistent search operation allows a client to receive notifications if an entry in the search scope is modified. If an entry is modified, the server sends a new copy of the entry to the LDAP client along and, if requested, an Entry Change Notification control that describes the change.

Beginning with Release 11g Release 1 (11.1.1.9.0), Oracle Internet Directory supports persistent search operations.

A persistent search operation uses the following controls:

- Persistent Search control - The LDAP client sends this control to the Oracle Internet Directory server with the persistent search request including specific options for returning the search results.
- Entry Change Notification control - If requested by the client, the server returns this control to the client for each changed entry that matches the search criteria in the search request.

Persistent searches are not be supported on special entries such as configuration entries, referrals, and entry aliases.

Persistent searches need to keep connections from the LDAP client to the server alive. Therefore, if a large number of persistent searches are issued, the LDAP connection limit might be reached and new connections could not be established. To prevent this situation from occurring, the `orclmaxpsearchconns` instance-specific attribute determines the maximum number of connections allowed for an LDAP persistent search operation.

An LDAP persistent search operation for Oracle Internet Directory follows this sequence:

1. The LDAP client authenticates to the Oracle Internet Directory server.

2. The client issues a persistent search request with an attached Persistent Search control. This control specifies the following information:
   - `changesOnly` - If this field is FALSE, the server returns the initial set of results that match the search criteria. Or, if the field is TRUE, the server does not return these entries.
   - `changeTypes` - This field specifies the type of changes the client wants to track. If subsequent changes are made to entries that match the search criteria, the server returns these changed entries to the client.

     The `changeTypes` field is a logical OR of one or more of these values:
     - add: 1
     - delete: 2
     - modify: 4
     - moddn: 8
   - `returnECs` - If this field is TRUE, the server also returns an Entry Change Notification control along with each changed entry.
3. The OID server processes the persistent search request using the options described in the previous step.

   The server does not return the `SearchResultDone` message to the client, because the connection must be kept alive until the client unbinds or abandons the operation.
4. If the server returns search results, the client processes these results as required.
5. The LDAP client ends the Persistent Search operation by sending an unbind or abandon request.

> **See Also:**
>
> - For information about the `orclmaxpsearchconns` attribute, see Table 9-1.
> - For information about the Persistent Search and Entry Change Notification controls, see Extensions to the LDAP Protocol in *Oracle Fusion Middleware Application Developer's Guide for Oracle Identity Management.*
> - Internet Engineering Task Force (IETF) document - Persistent Search: A Simple LDAP Change Notification Mechanism:
>
>   `http://tools.ietf.org/id/draft-ietf-ldapext-psearch-03.txt`

3.3 Understanding Directory Entries in Oracle Internet Directory

In an online directory, each collection of information about an object is called an entry. An entry can include, for example, information about an employee, a conference room, an e-commerce partner, or a shared network resource such as a printer.

This section contains the following topics:

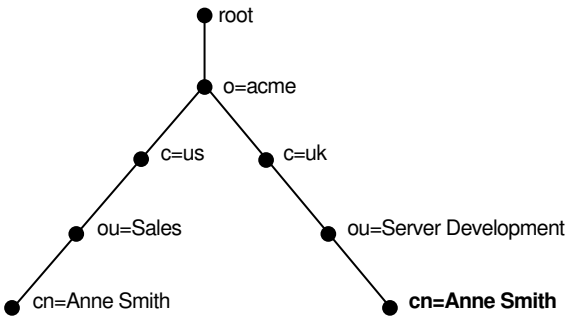
- Distinguished Names and Directory Information Trees
- Entry Caching

3.3.1 Distinguished Names and Directory Information Trees

Each entry in an online directory is uniquely identified by a distinguished name (DN). The distinguished name tells you exactly where the entry resides in the directory hierarchy. This hierarchy is represented by a directory information tree (DIT).

To understand the relation between a distinguished name and a directory information tree, look at Figure 3-3.

Figure 3-3 A Directory Information Tree



The DIT in Figure 3-3 includes entries for two employees of Acme Corporation who are both named Anne Smith. It is structured along geographical and organizational lines. The Anne Smith contained in the left branch works in the Sales division in the United States, while the other works in the Server Development division in the United Kingdom.

The Anne Smith contained in the right branch has the common name (`cn`) Anne Smith. She works in an organizational unit (`ou`) named Server Development, in the country (`c`) of United Kingdom of Great Britain and Northern Ireland (`uk`), in the organization (`o`) Acme.

The DN for this "Anne Smith" entry is:

```
cn=Anne Smith,ou=Server Development,c=uk,o=acme
```

Note that the conventional format of a distinguished name places the lowest DIT component at the left, then follows it with the next highest component, moving progressively up to the root.

Within a distinguished name, the lowest component is called the relative distinguished name (RDN). For example, in the previous entry for Anne Smith, the RDN is `cn=Anne Smith`. Similarly, the RDN for the entry immediately above Anne Smith's RDN is `ou=Server Development`, the RDN for the entry immediately above `ou=Server Development` is `c=uk`, and so on. A DN is thus a concatenation of RDNs that reflects parent-child relationships in the DIT. Within the DN, RDNs are separated by commas.

To locate a particular entry within the overall DIT, a client uniquely identifies that entry by using the full DN—not simply the RDN—of that entry. For example, within the global organization in Figure 3-3, to avoid confusion between the two Anne Smiths, you would use each one's full DN. If there are potentially two employees with the same name in the same organizational unit, you could use additional mechanisms—for example, you could identify each employee with a unique number.

3.3.2 Entry Caching

To make operations on entries quick and efficient, Oracle Internet Directory uses entry caching. When you enable this feature, Oracle Internet Directory assigns a unique identifier to each entry, then stores a specified number of those identifiers in cache memory. When a user performs an operation on an entry, the directory server looks in the cache for the entry identifier, then retrieves the corresponding entry from the directory.

Note:

- Beginning with Release 11*g* Release 1 (11.1.1.6.0), the entry cache resides in shared memory, so multiple Oracle Internet Directory server instances on the same host can share the same cache. Attributes for configuring the cache now reside in the DSA configuration entry.
- The entry cache is a write-through cache. An LDAP operation first writes to the database and then invalidates the corresponding cache entry. A subsequent search of that entry causes the cache to be refreshed.

See Also:

- See Server Entry Cache in *Tuning Performance*.
- Managing Knowledge References and Referrals.

3.4 Understanding the Concept of Attributes in Oracle Internet Directory

Understand about the conceptual description of attributes in online directory and their syntaxes, rules and options.

This section contains the following topics:

- Attributes in Online Directory
- Application and System Configuration Attributes
- Single-Valued and Multivalued Attributes
- Common LDAP Attributes
- Attribute Syntax
- Attribute Matching Rules
- Attribute Options

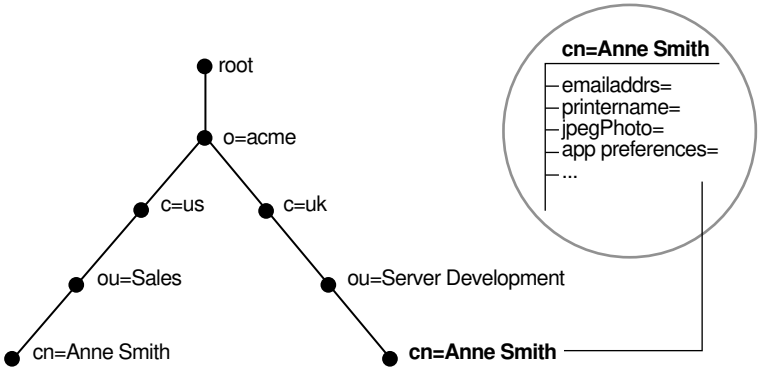
3.4.1 Attributes in Online Directory

In a typical telephone directory, an entry for a person contains such information items as an address and a phone number. In an online directory, such an information item is called an

attribute. Attributes in a typical employee entry can include, for example, a job title, an e-mail address, or a phone number.

For example, in Figure 3-4, the entry for Anne Smith in Great Britain (uk) has several attributes, each providing specific information about her. These are listed in the balloon to the right of the tree, and they include `emailaddrs`, `printername`, `jpegPhoto`, and `app preferences`. Moreover, each bullet in Figure 3-4 is also an entry with attributes, although the attributes for each are not shown.

Figure 3-4 Attributes of the Entry for Anne Smith



Each attribute consists of an attribute type and one or more attribute values. The attribute type is the kind of information that the attribute contains—for example, `jobTitle`. The attribute value is the particular occurrence of information appearing in that entry. For example, the value for the `jobTitle` attribute could be `manager`.

3.4.2 Application and System Configuration Attributes

Understand about the application and system configuration attributes.

Attributes contain two kinds of information.

- Application Attributes

  This information is maintained and retrieved by directory clients and is unimportant to the operation of the directory. A telephone number, for example, is application information.

- System Configuration Attributes

  This information pertains to the operation of the directory itself. Some operational information is specified by the directory to control the server—for example, the time stamp for the creation or modification of an entry, or the name of the user who creates or modifies an entry. Other operational information, such as access information, is defined by administrators and is used by the directory program in its processing.

> **See Also:**
>
> Managing System Configuration Attributes for more information about system configuration attributes.

3.4.3 Attributes Created with Each New Entry

To enhance your ability to search for entries, Oracle Internet Directory automatically creates several system configuration attributes when you add an entry to the directory.

These include:

Table 3-2 Attributes Created with Each New Entry

Attribute	Description
`creatorsName`	Name of the person creating the entry
`createTimestamp`	Time of entry creation in UTC (Coordinated Universal Time)
`modifiersName`	Name of person modifying the entry
`modifyTimestamp`	Time of last entry modification in UTC

Moreover, when a user modifies an entry, Oracle Internet Directory automatically updates the `modifiersName` and `modifyTimestamp` attributes to, respectively, the name of the person modifying the entry, and the time of the entry modification in UTC.

> **See Also:**
>
> Managing System Configuration Attributes for instructions on configuring system configuration attributes.

3.4.4 Single-Valued and Multivalued Attributes

Attributes can be either single-valued or multivalued. Single-valued attributes carry only one value in the attribute, whereas multivalued attributes can have several. An example of a multivalued attribute is a group membership list with names of everyone in the group.

3.4.5 Common LDAP Attributes

Oracle Internet Directory implements all of the standard LDAP attributes. Some of the more common ones are defined by RFC 2798 of the Internet Engineering Task Force (IETF).

See Table 3-3.

Table 3-3 Common LDAP Attributes

Attribute Type	Attribute String	Description
commonName	`cn`	Common name of an entry—for example, `Anne Smith`.
domainComponent	`dc`	The DN of the component in a Domain Name System (DNS)—for example, `dc=uk,dc=acme,dc=com`.
jpegPhoto	`jpegPhoto`	Photographic image in JPEG format. This is stored in binary format.
organization	`o`	Name of an organization—for example, `my_company`.
organizationalUnitName	`ou`	Name of a unit within an organization—for example, `Server Development`.

Table 3-3 (Cont.) Common LDAP Attributes

Attribute Type	Attribute String	Description
owner	`owner`	Distinguished name of the person who owns the entry, for example, `cn=Anne Smith, ou=Server Development, o= Acme, c=uk`.
surname, sn	`sn`	Last name of a person—for example, `Smith`.
telephoneNumber	`telephoneNumber`	Telephone number—for example, `(650) 123-4567` or `6501234567`.

> **See Also:**
>
> Oracle Identity Management LDAP Attribute Reference in *Reference for Oracle Identity Management* for a list of several attributes Oracle Internet Directory provides.

3.4.6 Attribute Syntax

Attribute syntax is the format of the data that can be loaded into each attribute. For example, the syntax of the `telephoneNumber` attribute might require a telephone number to be a string of numbers containing spaces and hyphens. However, the syntax for another attribute might require specifying whether the data has to be in the form of a date, or whether the data can consist of numbers only. Each attribute must have one and only one syntax.

Oracle Internet Directory recognizes most of the syntaxes specified in RFC 2252 of the Internet Engineering Task Force (IETF), allowing you to associate most of the syntaxes described in that document with an attribute. In addition to recognizing the syntaxes in RFC 2252, Oracle Internet Directory also enforces some LDAP syntaxes. You cannot add new syntaxes beyond those already supported by Oracle Internet Directory.

> **See Also:**
>
> About LDAP Attribute Syntax in *Reference for Oracle Identity Management*.

3.4.7 Attribute Matching Rules

In response to most incoming client requests, the directory server performs search and compare operations. During these operations, the directory server consults the relevant matching rule to determine equality between the attribute value sought and the attribute value stored.

For example, matching rules associated with the `telephoneNumber` attribute could cause "(650) 123-4567" to be matched with either "(650) 123-4567" or "6501234567" or both. When you create an attribute, you associate a matching rule with it.

Oracle Internet Directory implements all the standard LDAP matching rules. You cannot add new matching rules beyond those already supported by Oracle Internet Directory.

> **See Also:**
>
> - Managing Directory Schema for information on viewing matching rules.
> - About LDAP Matching Rules in *Reference for Oracle Identity Management*.

3.4.8 Attribute Options

An attribute type can have various options that enable you to specify how the value for that attribute is made available in a search or a compare operation.

For example, suppose that an employee has two addresses, one in London, the other in New York. Options for that employee's `address` attribute could allow you to store both addresses.

Moreover, attribute options can include language codes. For example, options for John Doe's `givenName` attribute could enable you to store his given name in both French and Japanese.

For clarity, we can distinguish between an attribute with an option and its base attribute, which is the same attribute without an option. For example, in the case of `givenName;lang-fr=Jean`, the base attribute is `givenName`; the French value for that base attribute is `givenName;lang-fr=Jean`.

An attribute with one or more options inherits the properties—for example, matching rules and syntax— of its base attribute. To continue the previous example, the attribute with the option `cn;lang-fr=Jean` inherits the properties of `cn`.

> **Note:**
>
> You cannot use an attribute option within a DN. For example, the following DN is incorrect: `cn;lang-fr=Jean, ou=sales,o=acme,c=uk`.

> **See Also:**
>
> Managing Directory Entries in Oracle Internet Directory.

3.5 Understanding Object Classes in Oracle Internet Directory

An object class is a group of attributes that define the structure of an entry. When you define a directory entry, you assign one or more object classes to it. Some of the attributes in these object classes are mandatory and must have values, others are optional and can be empty.

For example, the `organizationalPerson` object class includes the mandatory attributes `commonName` (`cn`) and `surname` (`sn`), and the optional attributes `telephoneNumber`, `uid`, `streetAddress`, and `userPassword`. When you define an entry by using the `organizationalPerson` object class, you must specify values for `commonName` (`cn`) and `surname` (`sn`). You do not need to provide values for `telephoneNumber`, `uid`, `streetAddress`, and `userPassword`.

This section contains the following topics:

- Subclasses, Superclasses, and Inheritance
- Understanding the Types of Object Class

3.5.1 Subclasses, Superclasses, and Inheritance

A subclass is an object class derived from another object class. The object class from which a subclass is derived is called its superclass.

For example, the object class `organizationalPerson` is a subclass of the object class `person`. Conversely, the object class `person` is the superclass of the object class `organizationalPerson`.

Subclasses inherit all of the attributes belonging to their superclasses. For example, the subclass `organizationalPerson` inherits the attributes of its superclass, `person`. Entries also inherit attributes that their superclasses have inherited.

> **Note:**
>
> In itself, an object class contains no values. Only an instance of an object class—that is, an entry—contains values. When a subclass inherits attributes from a superclass, it inherits only the attribute definitions of the superclass.

One special object class, called `top`, has no superclasses. It is one of the superclasses of every object class in the directory, and its attribute definitions are inherited by every entry.

3.5.2 Understanding the Types of Object Class

Understand about the various types of object classes available.

This section contains the following topics:

- Structural Object Classes
- Auxiliary Object Classes
- Abstract Object Classes

3.5.2.1 Structural Object Classes

Structural object classes describe the basic aspects of an object. Most of the object classes that you use are structural object classes, and every entry should belong to at least one structural object class. Examples of structural object classes are `person` and `groupOfNames`.

These object classes model real-world entities and their physical or logical attributes. Examples include people, printers, and database connections.

Structural object classes use structure rules to place restrictions on the kinds of objects you can create under any given object class. For example, a structure rule might require all objects below the `organization` (`o`) object class to be `organizational units` (`ou`). Following this rule, you could not enter `person` objects directly below an `organization` object class. Similarly, a structure rule might disallow you from placing an organizational unit (`ou`) object below a `person` object.

3.5.2.2 Auxiliary Object Classes

Auxiliary object classes are groupings of optional attributes that expand the existing list of attributes in an entry. Unlike structural object classes, they do not place restrictions on where an entry may be stored, and you can attach them to any entry regardless of that entry's location in the DIT.

> **Note:**
>
> Oracle Internet Directory does not enforce structure rules. It therefore handles both structural and auxiliary object classes in the same way.

3.5.2.3 Abstract Object Classes

An abstract object class is a virtual object class. It is used only for convenience when specifying the highest levels of the object class hierarchy. It cannot be the only object class for an entry. For example, the object class `top` is an abstract object class. It is required as a superclass for all structural object classes, but it cannot be used alone.

The `top` object class includes the mandatory attribute `objectClass` and several optional attributes. The optional attributes in `top` are:

- `orclGuid`: Global identification which remains constant if the entry is moved
- `creatorsName`: Name of the creator of the object class
- `createTimestamp`: Time when the object class was created
- `modifiersName`: Name of the last person to modify the object class
- `modifyTimestamp`: Time when the object class was last modified
- `orclACI`: access control list (ACL) directives that apply to all entries in the subtree below the access control policy point (ACP) where this attribute is defined
- `orclEntryLevelACI`: Access control policy pertaining to only a specific entity, for example, a special user

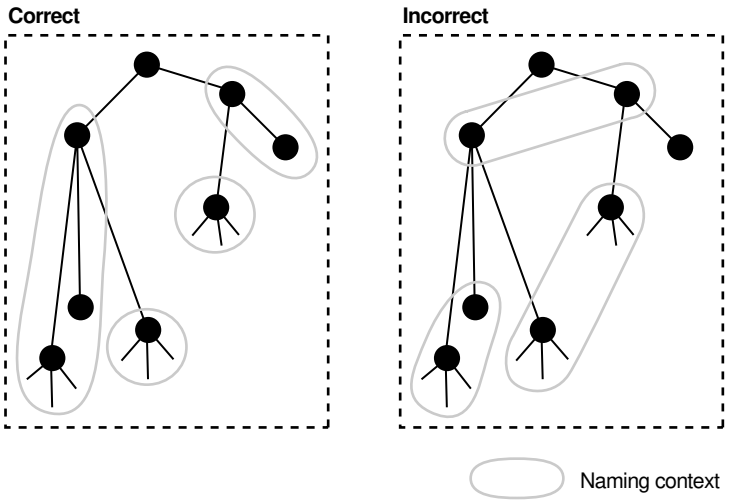
> **See Also:**
>
> – Globalization Support for more information on access control policies and ACLs
> – Understanding Attributes for a discussion of how to add additional content to entries

3.6 Directory Naming Contexts

A directory naming context is a subtree that resides entirely on one server. It must be a complete subtree, that is, it must begin at an entry that serves as the top of the subtree, and extend downward to either leaf entries or references to subordinate naming contexts. It can range in size from a single entry to the entire directory information tree (DIT).

Figure 3-5 illustrates correct and incorrect naming contexts. Notice that the correct ones on the left are contiguous, and the incorrect ones on the right are not.

Figure 3-5 Correct and Incorrect Naming Contexts



To enable users to discover specific naming contexts, you can publish those naming contexts in Oracle Internet Directory by using ldapmodify.

> **See Also:**
>
> Managing Naming Contexts in Oracle Internet Directory for instructions on how to publish a naming context

3.7 Security Features in Oracle Internet Directory

Oracle Internet Directory is a key element of Oracle Identity Management. You can deploy multiple Oracle components to work against a shared instance of Oracle Internet Directory. This sharing allows an enterprise to simplify security management across all applications.

In addition to the role it plays in the Oracle Identity Management infrastructure, Oracle Internet Directory provides many powerful features for protecting information.

Security features within Oracle Internet Directory itself include:

- The Secure Sockets layer: Ensuring that data is not modified, deleted, or replayed during transmission
- Data privacy: Ensuring that data is not inappropriately observed while it is stored in Oracle Internet Directory
- Password policies: Establishing and enforcing rules for how passwords are defined and used
- Authorization: Ensuring that a user reads or updates only the information for which that user has privileges
- Password protection: Ensuring that passwords are not easily discovered by others

- Authentication: Ensuring that the identities of users, hosts, and clients are correctly validated

You can use all these features to enforce a uniform security policy for multiple applications enabled for Oracle Internet Directory, and do so in either an enterprise or hosted environment. You do this by deploying the directory for administrative delegation. This deployment allows, for example, a global administrator to delegate to department administrators access to the metadata of applications in their departments. These department administrators can then control access to their department applications.

You can also use security features of the underlying Oracle Database, such as Transparent Data Encryption and Database Vault, to protect Oracle Internet Directory data.

> **See Also:**
>
> Advanced Administration: Security

3.8 Globalization Support

Oracle Internet Directory follows LDAP Version 3 internationalization (I18N) standards. These standards require that the database storing directory data use Unicode Transformation Format 8-bit (UTF-8) character set. With Oracle9*i*, Oracle added a new UTF-8 character set called AL32UTF8. This database character set supports the latest version of Unicode (3.2), including the latest supplementary characters. This allows Oracle Internet Directory to store the character data of almost any language supported by Oracle Globalization Support. Moreover, although several different application program interfaces are involved in the Oracle Internet Directory implementation, Oracle Internet Directory ensures that the correct character encoding is used with each API.

Globalization Support means support for both single-byte and multibyte characters. A single-byte character is represented by one byte of memory. ASCII text, for example, uses single-byte characters. By contrast, a multibyte character can be represented by more than one byte. Simplified Chinese, for example, uses multibyte characters. An ASCII representation of a simplified Chinese directory entry definition might look like this:

```
dn: o=\274\327\271\307\316\304,c=\303\300\271\372
objectclass: top
objectclass: organization
o: \274\327\271\307\316\304
```

Where the attribute values correspond to an ASCII representation of a simplified Chinese directory entry definition.

By default, the main Oracle Internet Directory components—OID Monitor (OIDMON), OID Control Utility (OIDCTL), Oracle directory server (OIDLDAPD), and Oracle directory replication server (OIDREPLD)—accept only the UTF-8 character set. The Oracle character set name is AL32UTF8.

The Oracle directory server and database tools are no longer restricted to run on a UTF8 database. However, you must ensure that all characters in the client character set are included in the database character set (with same or different character codes) if the database underlying the Oracle Internet Directory server is not AL32UTF8 or UTF8. Otherwise, there may be data loss during LDAP add, delete, modify, or modifydn operations if the client data cannot be mapped to the database character set.

Oracle Directory Services Manager uses Unicode (UTF-16—that is, fixed-width 16-bit Unicode). It can support internationalized character sets.

> **See Also:**
>
> - Globalization Support in the Directory .
> - Overview of Globalization Support in *Oracle Database Globalization Support Guide* for a detailed discussion of Globalization Support.

3.9 Understanding Distributed Directories

Although an online directory is logically centralized, it can be physically distributed onto several servers. This distribution reduces the work a single server would otherwise have to do, and enables the directory to accommodate a larger number of entries. A distributed directory can be either replicated or partitioned. When information is replicated, the same naming contexts are stored by more than one server. When information is partitioned, one or more unique, non-overlapping naming contexts are stored on each directory server. In a distributed directory, some information may be partitioned and some may be replicated.

This section contains the following topics:

- Directory Replication
- Directory Partitioning

3.9.1 Directory Replication

Replication is the process of copying and maintaining the same naming contexts on multiple directory servers.

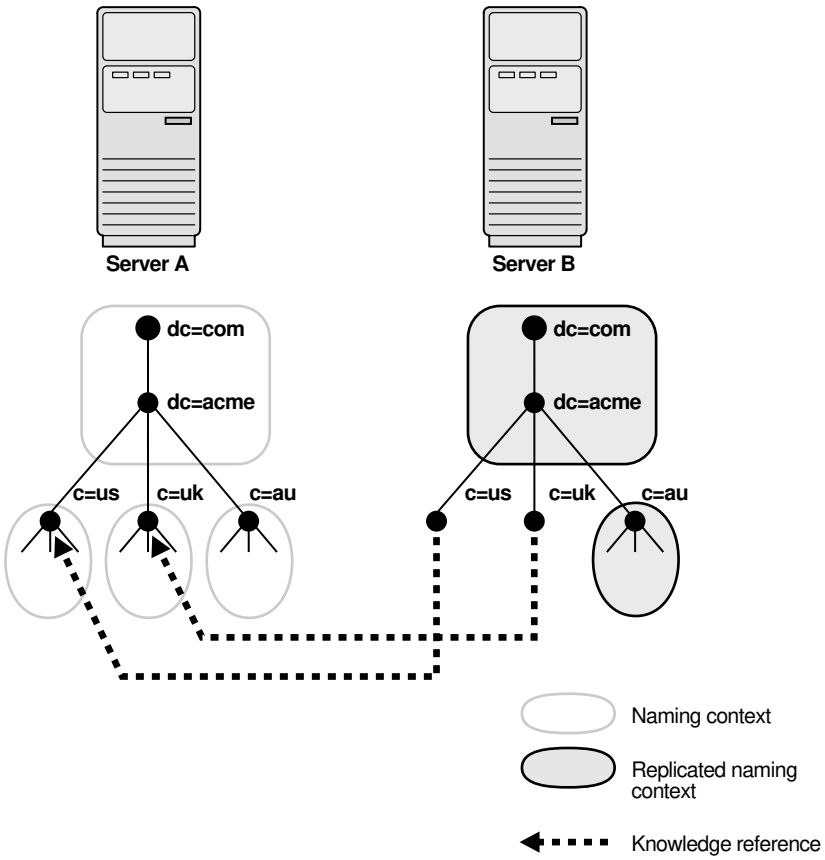
See Understanding Oracle Internet Directory Replication for a discussion of replication concepts.

3.9.2 Directory Partitioning

Partitioning, in which each directory server stores one or more unique, non-overlapping naming contexts, is another way of distributing directory information.

Figure 3-6 shows a partitioned directory in which some naming contexts reside on different servers.

Figure 3-6 A Partitioned Directory



In Figure 3-6, four naming contexts reside on Server A:

- `dc=acme,dc=com`
- `c=us,dc=acme,dc=com`
- `c=uk,dc=acme,dc=com`
- `c=au,dc=acme,dc=com`

Two naming contexts on Server A are replicated on Server B:

- `dc=acme,dc=com`
- `c=au,dc=acme,dc=com`

The directory uses one or more knowledge references to locate information that is requested of Server B, but that resides on Server A. It passes this information to a client in the form of a referral.

3.10 Knowledge References and Referrals

A knowledge reference provides the names and addresses of the various naming contexts held in another partition.

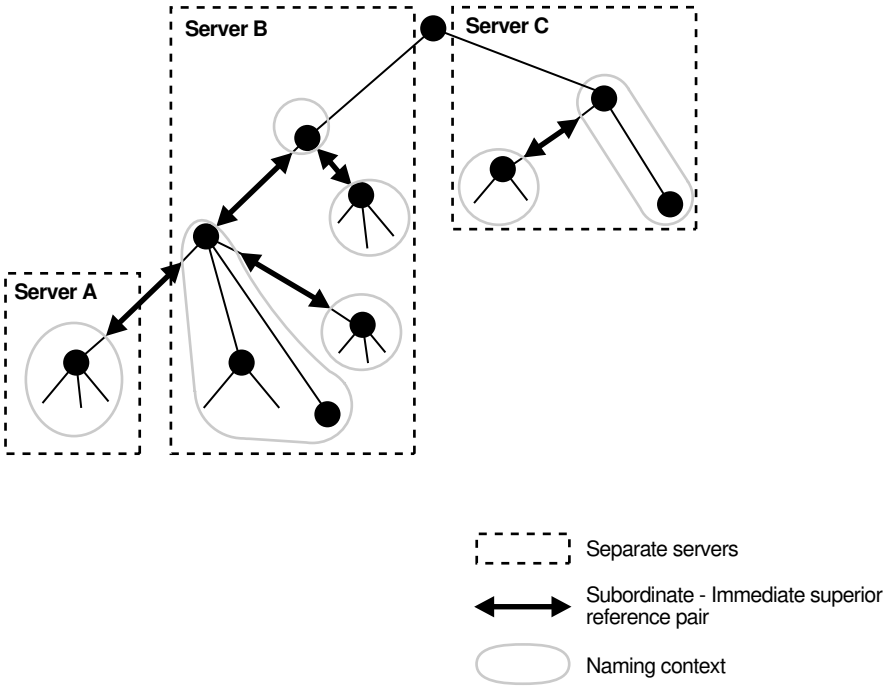
For example, in Figure 3-6, Server B uses knowledge references to point to the `c=us` and `c=uk` naming contexts on Server A. When Server B is asked for information residing on Server A, it

sends back one or more referrals to Server A. Clients can then use these referrals to contact Server A.

Typically, each directory server contains both superior and subordinate knowledge references. Superior knowledge references point upward in the DIT toward the root. They tie the partitioned naming context to its parent. Subordinate knowledge references point downward in the DIT to other partitions.

For example, in Figure 3-7, Server B holds four naming contexts, two of which are superior to the others. These two superior naming contexts use subordinate knowledge references to point to their subordinate naming contexts. Conversely, the naming context on Server A has an immediate superior residing on Server B. Server A therefore uses a superior knowledge reference to point to its parent on Server B.

Figure 3-7 Using Knowledge References to Point to Naming Contexts



Naming contexts that start at the top of the DIT obviously cannot have a knowledge reference to a superior naming context.

Note:

- There are presently no Internet standards for enforcing the validity of knowledge references, and Oracle Internet Directory does not do so. It is up to the administrator to ensure consistency among knowledge references within an enterprise network.
- Oracle recommends that permission for managing knowledge reference entries be restricted, as is the case with any other privileged administrative function such as schema or access control.

There are two kinds of referrals:

- Smart referrals

  These are returned to the client when the knowledge reference entry is in the scope of the search. It points the client to the server that stores the requested information.

  For example, suppose that:

  – Server A holds the naming context, `ou=server development,c=us,o=acme`, and has a knowledge reference to Server B.

  – Server B holds the naming context `ou=sales,c=us,o=acme`.

  When a client sends a request to Server A for information in `ou=sales,c=us,o=acme`, Server A provides the user with a referral to Server B.

- Default referrals

  These are returned when the base object is not in the directory, and the operation is performed in a naming context on another server. A default referral typically sends the client to a server that has more detailed information about the directory partitioning arrangement.

  For example, suppose that Server A holds:

  – The naming context `c=us,o=acme`

  – A knowledge reference to Server PQR that has more knowledge about the overall directory partitioning arrangement

  Now suppose that a client requests information on `c=uk,o=acme`. When Server A finds that it does not have the `c=uk,o=acme` naming context, it provides the client with a referral to Server PQR. From there, the client can find the server holding the requested naming context.

> **See Also:**
>
> Managing Knowledge References and Referrals.

3.11 Service Registry and Service to Service Authentication

The Service Registry and the Service to Service Authentication framework are Oracle Internet Directory features that facilitate integration between Oracle technology components that request services from one another. The Service Registry provides a place to store information, so that the components can discover each other. The Service to Service Authentication framework allows one component to authenticate to another and establishes trust among them.

The Service Registry is a container in Oracle Internet Directory (under cn=Services, Cn=OracleContext) where components store connectivity information, such as protocol, and other information, such as type of service. During installation, each OCS component registers its information in the Registry. At run-time the components discover information registered by other components. Service Registry objects are stored in the Oracle Internet Directory DIT in a component-specific Services container in the rootOracleContext.

Service to Service Authentication is a framework that allows one service to authenticate to another and establish trusts among the services. At install time, each of the client services is provisioned with a user name and password in Oracle Internet Directory. In addition, each

target service defines an authorization role in Oracle Internet Directory to control which components should it trust. When a component requests services of another component, the requestor must authenticate to the target service like any other client, using its own identity and credentials. The requesting service must also be listed in the Target services Trusted Application group (Default Group: contrasted Applications, counterpoise, cn=OracleContext). The requesting service also must send the user's identity so that the target service can authenticate the user as well. The data is sent securely, using either Digest authentication or the target service's native secure authentication.

3.12 Oracle Directory Integration Platform

Oracle Directory Integration Platform enables an enterprise to integrate its applications and other directories with Oracle Internet Directory. It provides all the interfaces and infrastructure necessary to keep the data in Oracle Internet Directory consistent with that in enterprise applications and connected directories. It also makes it easier for third-party vendors and developers to develop and deploy their own connectivity agents.

For example, an enterprise might want employee records in its HR database to be synchronized with Oracle Internet Directory. In addition, the enterprise may deploy certain LDAP-enabled applications (such as Oracle Portal) that must be notified whenever changes are applied to Oracle Internet Directory.

Based on the nature of integration, Oracle Directory Integration Platform provides two distinct services:

- The synchronization integration service, which keeps connected directories consistent with the central Oracle Internet Directory
- The provisioning integration service, which sends notifications to target applications to reflect changes made to a entries of interest, such as users and groups

> **See Also:**
>
> Synchronization, Provisioning, and the Differences Between Them in *Administering Oracle Directory Integration Platform*.

3.13 Understanding the Role of Identity Management in Oracle Internet Directory

Understand about the role of Identity Management and its realms in Oracle Internet Directory.

This section contains the following topics:

- Identity Management
- Understanding the Identity Management Realms

3.13.1 Identity Management

Identity management usually refers to the management of an organization's application users. Steps in their security life cycle include account creation, suspension, privilege modification, and account deletion. The managed entities may also include devices, processes, applications,

or anything else that must interact in a networked environment. They may also include users outside of the organization, for example customers, trading partners, or Web services.

Identity management is important to IT deployments because it can reduce administrative costs while at the same time improving security.

The Oracle Identity Management products enable deployments to manage centrally and securely all enterprise identities and their access to various applications in the enterprise. Identity management comprises these tasks:

- Creating enterprise identities and managing shared properties of these identities through a single enterprise-wide console
- Creating groups of enterprise identities
- Provisioning these identities in various services available in the enterprise. This includes:
  - Account creation
  - Account suspension
  - Account deletion
- Managing policies associated with these identities. These include:
  - Authorization policies
  - Authentication Policies
  - Privileges delegated to existing identities

> **Note:**
>
> For complete information and additional resources for Oracle Identity Management, go to the Oracle Technology Network (OTN) web site at `http://www.oracle.com/technetwork/index.html`.

3.13.2 Understanding the Identity Management Realms

Understand about the contextual description of Identity Management realm and Identity Management policies.

This section contains the following topics:

- Identity Management Realm
- Default Identity Management Realm
- Identity Management Policies

3.13.2.1 Identity Management Realm

An identity management realm defines an enterprise scope over which certain identity management policies are defined and enforced by the deployment. It comprises:

- A well-scoped collection of enterprise identities—for example, all employees in the US domain.
- A collection of identity management policies associated with these identities. An example of an identity management policy would be to require that all user passwords have at least one alphanumeric character.

- A collection of groups—that is, aggregations of identities—that simplifies the setting of the identity management policies.

You can define multiple identity management realms within the same Oracle Identity Management infrastructure. Multiple realms enable you to isolate user populations and enforce a different identity management policy—for example, password policy, naming policy, self-modification policy—in each realm.

Each identity management realm is uniquely named to distinguish it from other realms. It also has a realm-specific administrator with complete administrative control over the realm.

3.13.2.2 Default Identity Management Realm

For all Oracle components to function, an identity management realm is required. One particular realm, created during installation of Oracle Internet Directory, is called the default identity management realm. It is where Oracle components expect to find users, groups, and associated policies whenever the name of a realm is not specified.

There can be only one default identity management realm in the directory. If a deployment requires multiple identity management realms, then one of them must be chosen as the default.

3.13.2.3 Identity Management Policies

The Oracle Identity Management infrastructure supports a flexible set of management policies which comprise:

- Directory structure and naming policies that enable you to:
  - Customize the directory structure in Oracle Internet Directory for your deployment.
  - Specify where various identities are to be located and how they are uniquely identified.
- Authentication policies that enable you to specify authentication methods and protocols supported by the Oracle Identity Management infrastructure
- Identity management authorizations that enable you to control access to certain privileged services and delegate administration wherever necessary

> **Note:**
>
> In Oracle Internet Directory Release 9.0.2, the equivalent term for "identity management realm" was "subscriber."

3.14 TCP Keep-Alive Mechanism

By default, Oracle Internet Directory supports TCP keep-alive as long as the TCP keep-alive mechanism is enabled at the operating system level. No specific Oracle Internet Directory configuration is required.

However, if Oracle Virtual Directory is configured with Oracle Internet Directory server, TCP keep-alive is split between Listeners (what LDAP clients connect to) and Adapters (Oracle Internet Directory or other LDAP server). For this scenario, the following configuration is required:

- At the operating system level, enable the TCP keep-alive mechanism.

- For Oracle Virtual Directory, set the `vde.soTimeoutBackend` JVM parameter to specify how long Oracle Virtual Directory should wait for a response from Oracle Internet Directory before closing the connection. In other words, this parameter tells Oracle Virtual Directory to close inactive sockets after the time specified by the parameter.

  The `vde.soTimeoutBackend` parameter, if configured, helps Oracle Virtual Directory detect and safely close orphan server connections caused by remote client or server failure This greatly enhances the performance of the server.

  The value of `vde.soTimeoutBackend` parameter is passed to the Socket that is created using the Java API as follows:

  ```
  public void setSoTimeout(int timeout)
                  throws SocketException
  ```

  The preceding method allows you to enable or disable `SO_TIMEOUT` with the specified `timeout` limit parameter, in milliseconds. With this option set to a non-zero timeout, a read() call on the InputStream associated with this Socket will block for only this amount of time. If the timeout expires, a `java.net.SocketTimeoutException` is raised, though the Socket is still valid. The option must be enabled prior to entering the blocking operation to have effect. The timeout must be greater than `0`. A timeout of zero is interpreted as an infinite timeout.

3.15 Understanding the Concept of Resource Information

Understand about the contextual description or resource information components and where these components are located in the DIT from the following topics.

To fulfill the requests of users, some Oracle components gather data from various repositories and services. To gather the data, these components require the following information:

- Information specifying the type of resource from which the data is to be gathered. The type of resource could be, for example, an Oracle Database. This is called resource type information.
- Information for connecting and authenticating users to the resources. This is called resource access information.

These sections contains the following topics:

- Resource Type Information
- Resource Access Information
- Location of Resource Information in the DIT

3.15.1 Resource Type Information

Information about the resources that an application uses to service a user request is called resource type information. A resource type can be, for example, an Oracle Database or a Java Database Connectivity Pluggable Data Source. Resource type information includes such items as the class used to authenticate a user, the user identifier, and the password.

You specify resource type information by using the Oracle Internet Directory Self-Service Console.

3.15.2 Resource Access Information

Information for connecting and authenticating users to the databases is called resource access information. It is stored in an entry called a resource access descriptor (RAD) from which it can be retrieved and shared by various Oracle components.

For example, to service the request of a user for a sales report, Oracle Reports queries multiple databases. When it does this, it does the following:

1. Retrieves the necessary connect information from the RAD
2. Uses that information to connect to those databases and to authenticate the user requesting the data

After it has done this, it compiles the report.

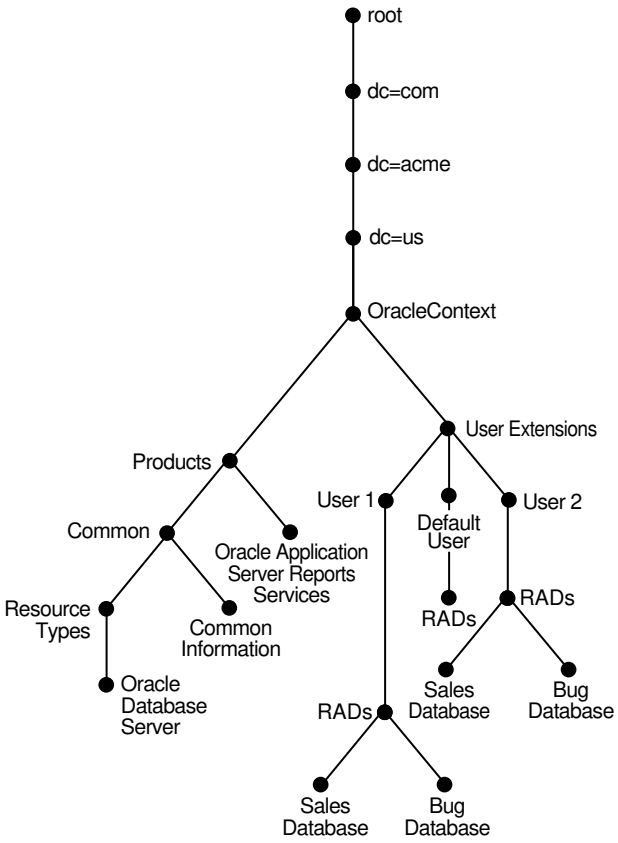
You specify resource access information by using the Oracle Internet Directory Self-Service Console. You can specify resource access information for each individual user or commonly for all users. In the latter case, all users connecting to a given application use, by default, the same information to connect to the necessary databases. Oracle recommends defining default resource access information whenever an application has its own integrated account management—for example, where each user is defined within the application itself with a unique single sign-on user name.

3.15.3 Location of Resource Information in the DIT

Understand how to locate resource information in DIT from the illustration and description provided below..

Figure 3-8 shows where resource information is located in the DIT.

Figure 3-8 Placement of Resource Access and Resource Type Information in the DIT



As Figure 3-8 shows, the resource access and resource type information is stored in the Oracle Context.

Resource access information for each user is stored in the `cn=User Extensions` node in the Oracle Context. In this example, the `cn=User Extensions` node contains resource access information for both the default user and for specific users. In the latter cases, the resource access information includes that needed for accessing both the Sales and the Bug databases.

Resource access information for each application is stored in the object identified by the application name—in this example, `cn=Oracle Reports Services, cn=Products,cn=Oracle Context,dc=us,dc=acme,dc=com`. This is the user information specific to that product.

Resource type information is stored in the container `cn=resource types, cn=common,cn=products,cn=Oracle Context`.

See Also:

- The sections about managing your own resource information, creating user entries, configuring default resources, and creating new resource types in *Reference for Oracle Identity Management* in the 14c library for instructions for an end user to specify resource access information
- Plug-in Schema Elements in *Reference for Oracle Identity Management*
- "Publishing Reports to the Web with Oracle Reports Services" in *Publishing Reports to the Web with Oracle Reports Services*

4

Understanding Process Control of Oracle Internet Directory Components

This chapter describes the Oracle Internet Directory process control architecture and related concepts for the LDAP server and the replication server processes. It includes starting, stopping, and monitoring the Oracle Internet Directory processes and some best practices for process control.

- Oracle Internet Directory Process Control Architecture
- The ODS_PROCESS_STATUS Table in Oracle Internet Directory
- Starting, Stopping, and Monitoring of Oracle Internet Directory Processes
- Oracle Internet Directory Replication-Server Control and Failover
- Oracle Internet Directory Process Control: Best Practices

For information on creating and destroying Oracle Internet Directory server instances, see Managing Oracle Internet Directory Instances.

For information on starting and stopping the Oracle Directory Integration Platform server refer to *Oracle Directory Integration Platform* .

4.1 Oracle Internet Directory Process Control Architecture

The Node Manager is a daemon process that monitors Oracle Fusion Middleware system components, including Oracle Internet Directory. Oracle Enterprise Manager Fusion Middleware Control uses Node Manager to stop or start instances of Oracle Internet Directory. If you stop or start Oracle Internet Directory components from the command line, you use `WLST`, the command-line interface to Node Manager.

See Managing Oracle Internet Directory Instances.

The Node Manager is responsible for the direct start, stop, restart and monitoring of the daemon process, OIDMON (*`$ORACLE_HOME`*`/bin/oidmon`). OIDMON is responsible for the process control of an Oracle Internet Directory instance. You can have multiple instances of Oracle Internet Directory on the same Oracle instance on the same node. The recommended way to create a new Oracle Internet Directory instance is to create a Oracle Fusion Middleware component of type OID. Each Oracle Internet Directory instance created in this manner has its own OIDMON.

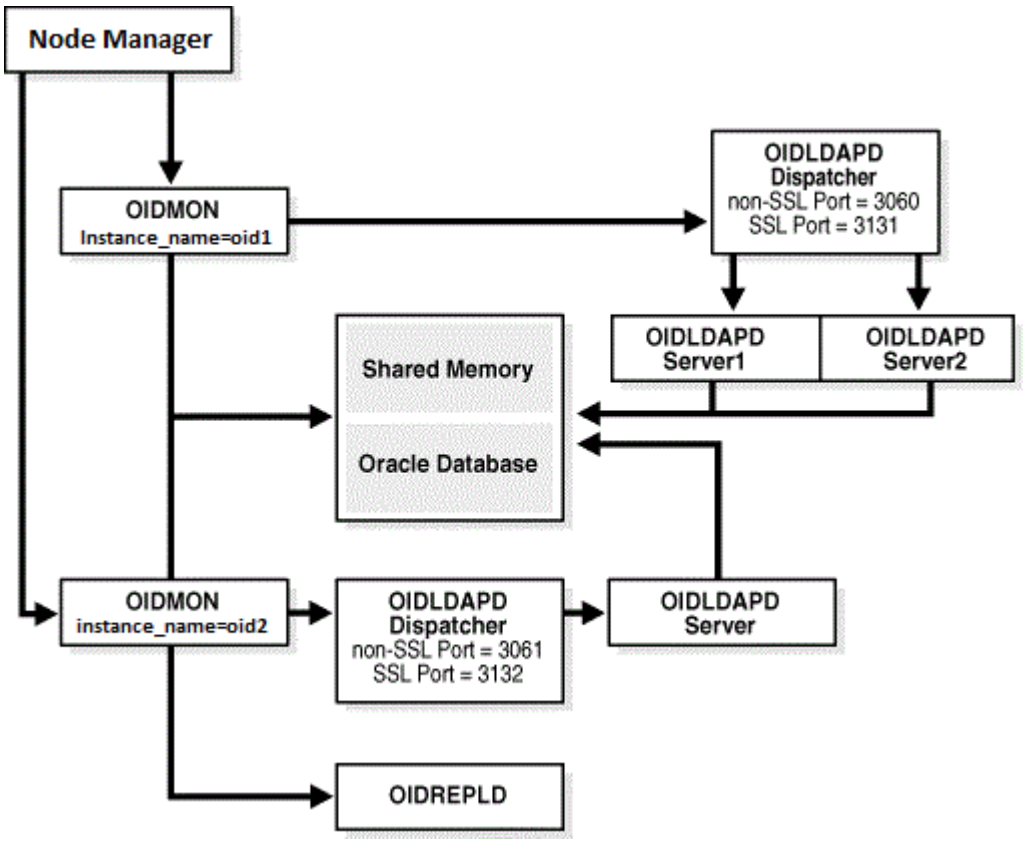
Figure 4-1 shows the overall architecture for Oracle Internet Directory process control. For each Oracle Fusion Middleware component of type OID, the Node Manager spawns an OIDMON process. The figure shows two components, `oid1` and `oid2`. OIDMON spawns the OIDLDAPD dispatcher process, then the dispatcher process spawns one or more OIDLDAPD server processes.

From Oracle Internet Directory 11g Release 1 (11.1.1.7.0) oward, the OIDLDAPD process is separated as the OIDDISPD (dispatcher) process and the OIDLDAPD (server) process. On UNIX and Linux systems, however, the `ps -ef` command will continue to show both of these processes as OIDLDAPD at runtime.

If replication is configured for that instance, OIDMON spawns a replication server process. Each dispatcher process has its own non-SSL and SSL port for receiving requests. The number of OIDLDAPD server processes that the dispatcher spawns for a component is controlled by the attribute `orclserverprocs` in the instance-specific configuration entry for the component.

> **Note:**
>
> If the log file size increases over `1` MB regardless the settings specified for `orclmaxlogfilesize` then OID Monitor (OIDMON) and OIDDISPD (dispatcher) log files are rotated after restarting Oracle Internet Directory.

Figure 4-1 Oracle Internet Directory Process Control Architecture



See Managing Oracle Internet Directory Instances for information about creating new Oracle Internet Directory instances.

4.2 The ODS_PROCESS_STATUS Table in Oracle Internet Directory

Oracle Internet Directory process information is maintained in the ODS_PROCESS_STATUS table in the `ODS` database user schema. OIDMON reads the contents of the table at a specified interval and acts upon the intent conveyed by the contents of that table. The interval is

controlled by the value of the `sleep` command line argument used at OIDMON startup, and the default value is 10 seconds.

Table 4-1 describes the information in the ODS_PROCESS_STATUS table that is relevant to process control:

Table 4-1 Process Control Items in the ODS_PROCESS_STATUS Table

Item	Meaning
Instance	Unique instance number for a given server ID on a given host
PID	Process ID of the server that is up and running
ServerID	Server ID (2=OIDLDAPD, 3=OIDREPLD)
Flags	Command line arguments that must be passed to the server instance
Hostname	Name of the host on which this server must be present
State	State of the Server Instance (0=stop, 1=start, 2=running, 3=restart, 4=shutdown, 5=failed-over, 7=delete, 8=add). OIDMON updates the state.
RetryCount	Number of attempts to start the server instance before it could be started successfully
Instancename	Name of the server instance, for example: server1
Compname	Name of the server component, for example: OID1

Note:

- There is a uniqueness constraint on: Instance, Instancename, Compname, ServerID, and Hostname.
- Details about ODS_PROCESS_STATUS are provided here for informational purposes only. Do not attempt to modify this table directly.

4.3 Starting, Stopping, and Monitoring of Oracle Internet Directory Processes

Understand about the events that occur when Node Manager starts and stops Oracle Internet Directory. It also describes process monitoring.

This section contains the following topics:

- Starting Oracle Internet Directory Using WLST Command
- Shutting Down Oracle Internet Directory Using WLST Command — shutdown ()
- Server Process Monitoring Using Node Manager

4.3.1 Starting Oracle Internet Directory Using WLST Command

Understand the process of starting an Oracle Internet Directory using WebLogic Scripting Tool (WLST) commands.

To start Oracle Internet Directory, follow the below given procedure:

1. You start an Oracle Internet Directory instance with Oracle Enterprise Manager Fusion Middleware Control or with the WLST command `start()`.
2. Node Manager issues an `oidmon start` command with appropriate arguments.
3. OIDMON then starts all Oracle Internet Directory Server instances whose information in the ODS_PROCESS_STATUS table has `state` value 1 or 4 and `DOMAIN_HOME`, `COMPONENT_NAME`, `INSTANCE_NAME` values matching the environment parameters set by the Node Manager.

See Also:

- Starting the Oracle Internet Directory Server by Using Fusion Middleware Control
- Starting the Oracle Internet Directory Server by Using WLST Command — start()

4.3.2 Shutting Down Oracle Internet Directory Using WLST Command — shutdown ()

Understand the process and prerequisites to shut down an Oracle Internet Directory using WebLogic Scripting Tool (WLST) commands.

Note:

- Before executing the `shutdown()` command, ensure that you connect to the weblogic server by using the `connect` command.
  The syntax for connecting to weblogic admin server is:

  ```
  connect(username='weblogic',password='weblogic-password',url='t3://
  admin-server-host:admin-server-port')
  ```

- Ensure that the Node Manager is up and running on the machine where you want to shutdown Oracle Internet Directory instance.
- Alternatively, you can shutdown Oracle Internet Directory instance using `stopComponent.sh` command. Before executing `stopComponent.sh` command, ensure that the Node Manager is up and running. You need not connect to WebLogic Server to execute `stopComponent.sh` command. The syntax for `stopComponent.sh` is:

  ```
  $DOMAIN_HOME/bin/stopComponent.sh <instance-name>
  ```

To Shutdown the Oracle Internet Directory instance, type:

```
shutdown(name='instance-name'))
```

4.3.3 Server Process Monitoring Using Node Manager

Node Manager does not monitor server processes directly. The Node Manager monitors OIDMON and OIDMON monitors the server processes.

The events are as follows:

- When you start OIDMON through Node Manager, the Node Manager starts OIDMON and ensures that OIDMON is up and running.
- If OIDMON goes down for some reason, Node Manager brings it back up.
- OIDMON monitors the status of the Oracle Internet Directory dispatcher process, LDAP server processes, and replication server process and makes this status available to Node Manager and Fusion Middleware Control.

4.4 Oracle Internet Directory Replication-Server Control and Failover

Understand the process control and failover for the Oracle Internet Directory replication server (oidrepld) in a Maximum Availability Architecture (MAA).

This section contains the following topics:

- Understanding the OID Monitor and Replication Server Failover
- Enabling Failover for Oracle Internet Directory Processes

4.4.1 Understanding the OID Monitor and Replication Server Failover

On each node, the Node Manager is responsible for the direct start, stop, restart, and monitoring of the OID Monitor process (oidmon). The OID Monitor is then responsible for the process control of an Oracle Internet Directory component, including the LDAP server processes and the replication server (oidrepld) process, if replication is configured for the instance.

In a Maximum Availability Architecture (MAA), these processes run as follows:

- Multiple OID Monitor processes can be running on a node and can connect to the same Oracle database. Each OID Monitor is responsible for an Oracle Internet Directory component, such as oid1 and oid2. However, only one OID Monitor performs high availability operations.
- If replication is configured, the replication server process should be running on only one node at any given time.

Every 60 seconds, the OID Monitor reports that it is running by sending a message to the Oracle database. The OID Monitor also polls the database to determine if any of the other nodes have failed. The OID Monitor uses the ODS_PROCESS_STATUS and ODS_SHM tables to communicate the starting and stopping of processes and the metadata changes across the nodes.

On the node running the replication server, the OID Monitor detects if the replication server is not responding and then tries to restart it. If the restart fails, the replication server is started on one of the surviving nodes, using the following logic:

- Pull logic: If an OID Monitor detects that another OID Monitor is down on another node for more than the configured time (specified by the `orclfailoverenabled` attribute), then the OID Monitor treats that node as failed and starts the replication server on its node.
- Push logic: The local OID Monitor on a node can push a failed replication server to another node by requesting that the OID Monitor on the other node start the process. An OID Monitor cannot start a replication server process on another node.

Later, if the failed node comes back up and the OID Monitor is restarted on that node, the OID Monitor will detect that another node is running the replication server. The OID Monitor will then request that the OID Monitor on the other node stop the replication server so that it can start the replication server on the original node.

4.4.2 Enabling Failover for Oracle Internet Directory Processes

The `orclfailoverenabled` attribute is a configuration entry (`"cn=configset,cn=oidmon,cn=subconfigsubentry"`) that configures failover for Oracle Internet Directory processes.

This attribute specifies the failover time in minutes before the OID Monitor will start failed processes on a surviving node. The default failover time is 5 minutes. A value of zero (0) specifies that Oracle Internet Directory processes will not fail over to another node.

For example, if the OID Monitor on NodeB detects that NodeA has not responded for the time specified by `orclfailoverenabled`, NodeB considers NodeA as having failed. NodeB then retrieves the necessary information from the Oracle database about the Oracle Internet Directory processes that were running on NodeA and tries to start them on NodeB.

To check the `orclfailoverenabled` attribute for the current failover time:

```
ldapsearch -h oid_host  p oid_port -D "cn=orcladmin" -q -s base \
-b "cn=configset,cn=oidmon,cn=subconfigsubentry" objectclass=* orclfailoverenabled
```

To specify a different failover time, set the value of the `orclfailoverenabled` attribute. For example, to set the failover time to 10 minutes:

```
ldapmodify -h oid_host  p oid_port -D "cn=orcladmin" -q -f enablerepl.ldif
```

where enablerepl.ldif contains:

```
dn: cn=configset,cn=oidmon,cn=subconfigsubentry
changetype: modify
replace: orclfailoverenabled
orclfailoverenabled: 10
```

4.5 Oracle Internet Directory Process Control: Best Practices

Learn about the best practices for using `wlst` and `oidctl`.

The recommended approach is as follows:

- Before you use the `startComponent.sh` or `stopComponent.sh` to start or stop the Oracle Internet Directory, you must set the `WLST_PROPERTIES` depending on the secured production mode that you have configured.

  Using Secured Production Mode with Demonstration Keystores with KSS

  Before invoking `wlst`, you may need to perform additional configuration to support the use of the demo keystores (included by default with WebLogic Server) provided through the

Keystore Service in domains with secured production mode enabled. Update the `WLST_PROPERTIES` environment variable to configure keystores as follows:

```
setenv WLST_PROPERTIES "-
Dweblogic.security.SSL.trustedCAKeyStore=<WL_HOME>/server/lib/
DemoTrust.jks -Dweblogic.ssl.AcceptKSSDemoCertsEnabled=true"
```

Using Secured Production Mode with Demonstration Keystores

Before invoking `wlst`, you may need to perform additional configuration to support the use of the demo keystores in domains with secured production mode enabled.

1. Set `UseKSSForDemo=false` in `<DOMAIN_HOME>/nodemanager/nodemanager.properties` file.
2. Update the `WLST_PROPERTIES` environment variable to configure keystores as follows:

   ```
   setenv WLST_PROPERTIES "-Dweblogic.RootDirectory=<DOMAIN_HOME> -
   Dweblogic.security.TrustKeyStore=DemoTrust -
   Dweblogic.security.SSL.ignoreHostnameVerification=true"
   ```

Using Secured Production Mode with Custom Keystores

Before invoking `wlst`, you may need to perform additional configuration to support the use of the custom keystores with secured production mode enabled. Update the `WLST_PROPERTIES` environment variable to configure keystores as follows:

1. Update the `nodemanager.properties` file with the following attributes and their values:

   ```
   CustomIdentityAlias
   CustomIdentityKeyStoreFileName
   CustomIdentityPrivateKeyPassPhrase
   CustomIdentityKeyStorePassPhrase
   KeyStores
   ```

2. Update the `WLST_PROPERTIES` environment variable to configure keystores as follows:

   ```
   setenv WLST_PROPERTIES "-Dweblogic.security.TrustKeyStore=CustomTrust -
   Dweblogic.security.CustomTrustKeyStoreFileName=trustKeystoreFile -
   Dweblogic.security.CustomTrustKeyStorePassPhrase=trustKeyStorePassword"
   ```

- Use `wlst` to stop or start Oracle Internet Directory as a component. That is, use it to stop or start all Oracle Internet Directory LDAP and replication server instances.
  - Using `stopComponent.sh` to stop Oracle Internet Directory causes Node Manager to issue an `oidmon stop`, which results in OIDMON shutting down all configured LDAP and replication server instances.
  - Using `startComponent.sh` to start Oracle Internet Directory causes Node Manager to issue an `oidmon start`, which results in OIDMON starting up all configured LDAP and replication server instances.

> **See Also:**
>
> – Managing Oracle Internet Directory Instances
> – Setting Up Replication for more information on starting and stopping Oracle Internet Directory

- Use `oidctl` to stop and start an Oracle Internet Directory Replication Server Instance without affecting the associated OIDMON and LDAP server instance managed by OIDMON.

> **See Also:**
>
> – Setting Up Replication for more information on starting and stopping a replication server.
> – Managing Oracle Internet Directory Instances by Using OIDCTL .

5

Understanding Oracle Internet Directory Organization

This chapter describes the considerations when you are designing the logical organization of the directory information for Oracle Internet Directory, including planning the directory information tree (DIT), the overall directory structure, and the names and organization of users and groups. It also describes migrating a DIT from a third-party directory.

- Understanding Directory Information Tree
- How to Plan the Overall Directory Structure
- Planning the Names and Organizing Users and Groups
- About DIT View Since 11g Release
- Migration of DIT from a Third-Party Directory

5.1 Understanding Directory Information Tree

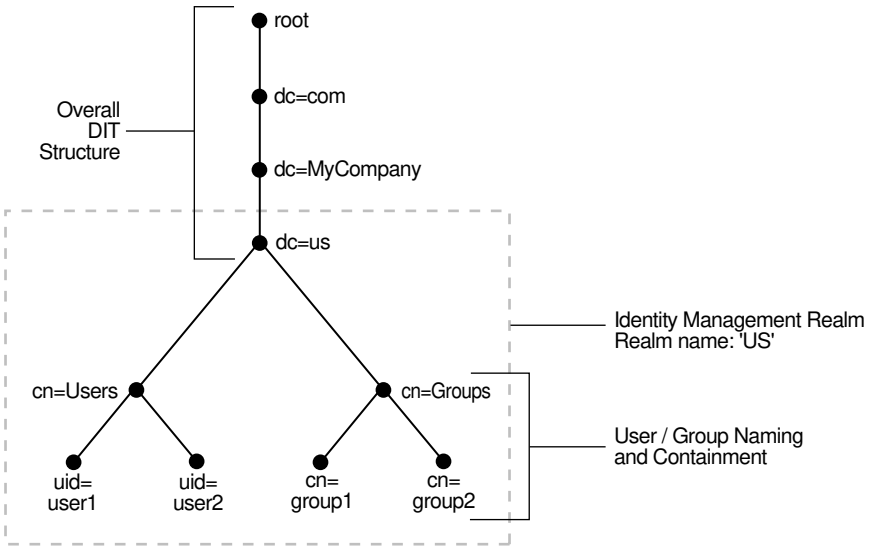
Oracle Internet Directory serves as a shared repository for the entire Oracle Identity Management infrastructure.

A carefully planned logical structure of the directory enables:

- Enforcement of security policies that meet the requirements of your deployment
- An efficient physical deployment of the directory service
- Easier configuration of synchronization of a third-party directory with Oracle Internet Directory

Figure 5-1 shows a directory information tree for a hypothetical company, called MyCompany, that is deploying identity management.

Figure 5-1 Planning the Directory Information Tree



MyCompany makes the following decisions with respect to the logical organization of the directory in their U.S. deployment:

- A domain name-based scheme is to represent the overall DIT hierarchy. Because the identity management infrastructure is being rolled out in the `us` domain, the root of the DIT is `dc=us,dc=mycompany,dc=com`.
- Within the naming context chosen, all users are represented under a container called `cn=users`. Within this container, all users are represented at the same level—that is, there is no organization-based hierarchy. In addition, the `uid` attribute is chosen as the unique identifier for all users.
- Within the naming context chosen, all enterprise groups are represented under a container called `cn=groups`. Within this container, all enterprise groups are represented at the same level. The naming attribute for all group entries is `cn`.
- Finally, the container `dc=us` is chosen as the root of the identity management realm. In this case, the name of the realm is `us`. The deployment expects to enforce similar security policies for all users who fall under the scope of the `us` realm.

Planning the logical organization of the directory for Oracle Identity Management involves planning:

- The overall structure of the directory information tree
- The directory containment and naming for users and groups
- The identity management realm

5.2 How to Plan the Overall Directory Structure

This task involves designing the basic directory information tree that all identity management-integrated applications in the enterprise are to use.

As you do this, keep these considerations in mind:

- The directory organization should facilitate clean and effective access control. If either full or partial replication is planned, then proper boundaries and policies for replication can be enforced only if the design of the DIT brings out the separation.
- If the enterprise is integrating with another directory server, it is best to align the DIT design of Oracle Internet Directory with the existing DIT.

  Other directory servers include Oracle products such as Oracle Unified Directory and Oracle Directory Server Enterprise Edition or third-party directory servers such as Microsoft Active Directory.

  This consideration also applies to deployments that are rolling out Oracle Internet Directory now but plan to roll out another directory server later—for example, Active Directory, which is required for the operation of software from Microsoft.

  In both cases, choosing an Oracle Internet Directory DIT design that is consistent with the integrated directory server makes the management of user and group objects easier through Oracle Identity Management and other middle-tier applications.
- In a single enterprise scenario, choosing a DIT design that aligns with the DNS domain name of the enterprise suffices. For example, if Oracle Internet Directory is set up in a company having the domain name `mycompany.com`, then a directory structure that has `dc=mycompany,dc=com` is recommended. Oracle recommends that you not use departmental or organization level domain components such as `engineering` in `engineering.mycompany.com`.
- If the enterprise has an X.500 directory service, and no other third-party LDAP directories in production, then it may benefit by choosing a country-based DIT design. For example, a DIT design with the root of `o=mycompany, c=US` might be more suitable for enterprises which already have an X.500 directory service.
- Because the directory can be used by several applications—both from Oracle and from third-parties alike—the naming attributes used in relative distinguished names (RDNs) constituting the overall DIT structure should be restricted to well-known attributes. The following attributes are generally well-known among most directory-enabled applications:
  - `c`: The name of a country
  - `dc`: A component of a DNS domain name
  - `l`: The name of a locality, such as a city, county or other geographic region
  - `o`: The name of an organization
  - `ou`: The name of an organizational unit
  - `st`: The name of a state or province
- A common mistake is to design the DIT to reflect either the corporate divisional or organizational structure. Because most corporations undergo frequent reorganization and divisional restructuring, this is not advisable. It is important to insulate the corporate directory from organizational changes as much as possible.

5.3 Planning the Names and Organizing Users and Groups

Understand about what to consider when modeling users and groups in Oracle Internet Directory. Most of the design considerations that are applicable to the overall DIT design are also applicable to the naming and containment of users and groups.

This section contains the following topics:

- Organizing Users in Oracle Internet Directory

- Organizing Groups in Oracle Internet Directory

5.3.1 Organizing Users in Oracle Internet Directory

The Oracle Identity Management infrastructure uses Oracle Internet Directory as the repository for all user identities. Even though a user might have account access to multiple applications in the enterprise, there is only one entry in Oracle Internet Directory representing that user's identity.

The location and content of these entries in the overall DIT must be planned before deploying Oracle Internet Directory and other components of the Oracle Identity Management infrastructure.

- As mentioned in the previous section: How to Plan the Overall Directory Structure, it is tempting to organize users according to their current departmental affiliations and hierarchy. However, this is not advisable because most corporations undergo frequent reorganization and divisional restructuring. It is more manageable to capture a person's organizational information as an attribute of that person's directory entry.
- There are no performance benefits derived from organizing users in a hierarchy according to organizational affiliations or management chain. Oracle recommends that you keep the DIT containing users as flat as possible.
- If the deployment has different user populations with each one maintained and managed by a different organization, then Oracle recommends subdividing users into containers based on these administrative boundaries. This simplifies the setting of access controls and helps in cases where replication is needed.
- The out-of-the-box default nickname attribute for uniquely identifying users in lookup operations is `uid`. This is the default attribute used for logins. The out-of-the-box default naming attribute for constructing a DN is `cn`.
- The default attribute for uniquely identifying users is cn or CommonName. The typical value of CommonName is the person's full name. People's names or email addresses, however, can change and therefore might not be suitable as values of this attribute. If possible, choose a value that will not change and that uniquely identifies a user, such as the employee id.
- Typically, most enterprises have a Human Resources department that establishes rules for assigning unique names and numbers for employees. When choosing a unique naming component for directory entries, it is good to exploit this administrative infrastructure and use its policies.
- It is required that all user entries created in the directory belong to the following object classes: `inetOrgPerson`, `orclUserV2`.
- If you already have a third-party directory, or plan to integrate with one in the future, then it is beneficial to align the user naming and directory containment in Oracle Internet Directory with the one used in the third-party directory. This simplifies the synchronization and subsequent administration of the distributed directories.

> **Note:**
>
> In Oracle Internet Directory Release 9.0.2, the default value for the `nickname` attribute was `cn`. After Release 9.0.4, the default value for this attribute is `uid`.

5.3.2 Organizing Groups in Oracle Internet Directory

Some applications integrated with the Oracle Identity Management infrastructure can also base their authorizations on enterprise-wide groups created by the deployment in Oracle Internet Directory. Like user entries, the location and content of these group entries should also be carefully planned.

When you design groups, consider the following:

- There are no performance benefits to be gained from organizing enterprise groups in a hierarchy based on the organizational affiliations or ownership. Oracle recommends keeping the DIT that contains groups as flat as possible. This facilitates easy discovery of groups by all applications and fosters sharing of these groups across applications.
- It is preferable to separate the users and groups in the DIT so that different management policies can be applied to each set of entries.
- The attribute used to uniquely identify a group should be `cn` or `CommonName`.
- All group entries created by the enterprise in the directory should belong to the following object classes: `groupOfUniqueNames` and `orclGroup`. The former object class is an internet standard for representing groups. The latter is useful when using the Oracle Internet Directory Self-Service Console to manage groups.
- Instead of creating new directory access controls for each enterprise-wide group, consider doing the following:
  1. Use the `owner` attribute of the group to list which users own this group.
  2. Create an access control policy at a higher level that grants all users listed in the `owner` attribute special privileges to perform the various operations.
- In the `description` attribute, provide information for users to understand the purpose of the group.
- Consider using the `displayName` attribute from the `orclGroup` object class. This attribute enables applications to display a more readable name for the group.
- If you have different sets of groups, each of which is maintained and managed by a different organization with its own administrative policies, then sub-divide the groups into containers based on these administrative boundaries. This simplifies the setting of access controls. It also helps when replication is needed.
- If you already have a third-party directory, or plan to integrate with one in the future, then align the group naming and directory containment in Oracle Internet Directory with the one used in the third-party directory. This simplifies the synchronization and subsequent administration of the distributed directories.

5.4 About DIT View Since 11g Release

From 11g Release 1 (11.1.1.9.0) onward, you can specify a DIT view, which is a virtual view or namespace that shows entries from a different or source DIT. A DIT view reduces the overall storage footprint in a cloud environment for common entries across tenants.

In a DIT view, the following items are transformed from the source DIT to the target DIT view namespace:

- DN and all DN attributes in the entry
- RDN mappings that are specified for the DIT view

For more information, see Creating a DIT View.

5.5 Migration of DIT from a Third-Party Directory

Understand how to migrate from a Third-Party Directory.

To migrate a DIT from a third-party directory, use the techniques described in Migrating Data from Other Data Repositories and Synchronizing With Third-party Meta Directory Solutions. in *Administering Oracle Directory Integration Platform* .

If you are migrating a DIT from a Microsoft Active Directory environment, also see the chapter on integration with the Microsoft Active Directory Environment. Oracle recommends that you configure the Oracle Internet Directory DIT to be identical to the third-party DIT.

6

Understanding Oracle Internet Directory Replication

This chapter provides an introduction to Oracle Internet Directory replication, Replication concepts and various replication methods that can be used based on your organization's requirement.

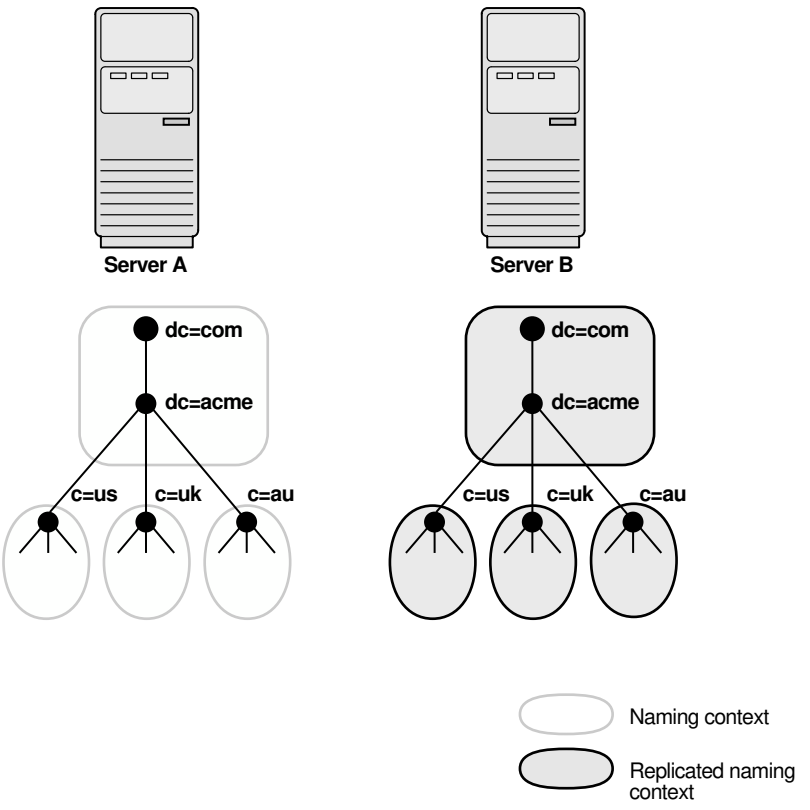
- Why Use Oracle Internet Directory Replication?
- Understanding Basic Concepts of Internet Directory Replication
- What Kind of Replication Do You Need?

6.1 What is Oracle Internet Directory Replication?

Replication is the process of copying and maintaining the same naming contexts on multiple directory servers. Replication can improve performance by providing more servers to handle queries and by bringing the data closer to the client. Replication can also improve reliability by eliminating risks associated with a single point of failure.

Figure 6-1 shows a replicated directory.

Figure 6-1 A Replicated Directory



See Also:

- Advanced Administration: Directory Replication
- How Replication Works

6.2 Why Use Oracle Internet Directory Replication?

Understand the use of Oracle Internet Directory Replication.

There are many reasons to implement replication, including the following:

- Local accessibility and performance requirements

  Most corporations have operations in many regions in the world, and those operations need a common directory. Suppose that the regions were interconnected with low bandwidth links involving multiple intermediate routers. A client accessing a directory server from outside the region could experience a very high latency, and even inadequate throughput. In such cases, a regional replica is essential.

- Load balancing

  When directory access exceeds the capacity of an existing server, an additional server can share the load. Even if the deployment meets the load, it can be less costly to maintain two relatively low-end systems than one high-end system.

- Failure tolerance and higher overall system availability

  One of the most important reasons to implement directory replication is to increase overall system availability. When one server is unavailable, the traffic can be routed to other available servers. This can be transparent to clients.

6.3 Understanding Basic Concepts of Internet Directory Replication

Get introduced to some of the basic concepts of Internet Directory Replication.

This section contains the following topics:

- How to Decide Full or Partial Content Replication?
- Replication Direction: One-Way, Two-Way, or Peer to Peer
- Transport Mechanism: LDAP
- Example for Single-master, Multimaster, or Fan-out Directory Replication Group Type
- Loose Consistency Model in Directory Replication Architecture
- Multimaster Replication with Fan-Out

6.3.1 How to Decide Full or Partial Content Replication?

When setting up replication, you must decide how much of the DIT to replicate from one node to another.

The choices are:

Table 6-1 Full or Partial Replication

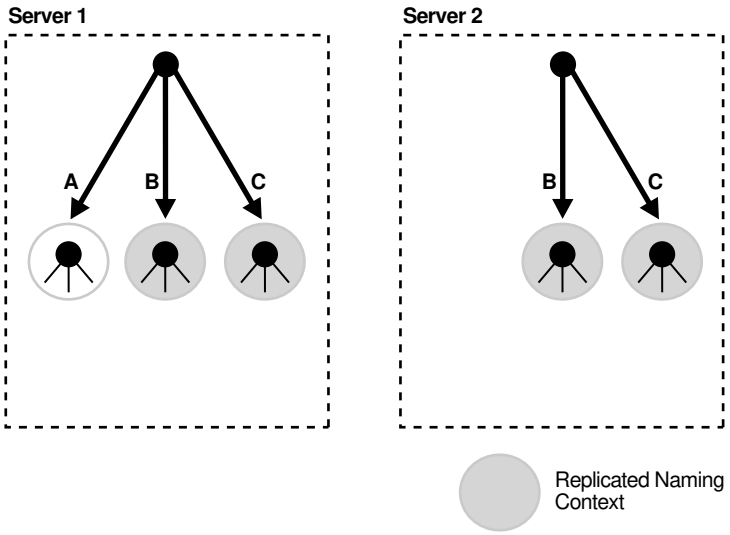
Content to Replicate	Description
Full	Propagates the entire DIT to another node.
Partial	Propagates one or more subtrees, rather than the entire DIT, to another node.

Full replication involves propagating the entire DIT to another node. This type of replication ensures the high availability of the entire directory. You can also use it to distribute operations on the entire directory among different nodes. Full replication is based on LDAP.

Partial replication enables you to propagate one or more subtrees, rather than the entire DIT, to another node. Decentralizing a directory in this way enables you to balance the workload between servers and build a highly available distributed directory, complete with fault tolerance and failover. You can set up partial replication by using command-line tools. Partial replication is based on LDAP-based protocol.

Figure 6-2 shows an example of partial replication.

Figure 6-2 Example of Partial Replication



6.3.2 Replication Direction: One-Way, Two-Way, or Peer to Peer

Understand about the different replication directions.

The direction of replication can be one-way, two-way or peer-to-peer:

Table 6-2 Direction of Replication

Direction	Description
One-Way	One node is configured as the supplier and the other as the consumer. The consumer is read-only.

Table 6-2 (Cont.) Direction of Replication

Direction	Description
Two-Way	Both nodes are both supplier and consumer. They are both read/write, or updatable.
Peer-to-Peer	All nodes in a replication group are both supplier and consumer to all other nodes

Sometimes the terms read-only and read/write are used to describe direction of replication. In a one-way replication agreement, the consumer node is said to be read-only. That is, you cannot propagate changes to other nodes by writing to that node. In a two-way replication agreement, both nodes are considered to be read/write. Another term for read/write is updatable.

Sometimes the term multimaster is used instead of peer-to-peer. Multimaster actually refers to the type of replication group, as described in Example for Single-master, Multimaster, or Fan-out Directory Replication Group Type.

6.3.3 Transport Mechanism: LDAP

Oracle Internet Directory supports LDAP protocol that can be used for replicating data from one node to another.

Table 6-3 Transport Protocols

Transport Mechanism	Description
LDAP	Uses the industry-standard Lightweight Directory Access Protocol Version 3. This is the recommended protocol to use for replication, unless you are also performing Oracle Single Sign-On replication.

LDAP replication can be configured as peer-to-peer, one-way or two-way.

> **Note:**
>
> Oracle Database Advanced Replication is deprecated from 12.2.1.3.0 and is no longer in use.

6.3.4 Directory Replication Group (DRG) Type: Single-master, Multimaster, or Fan-out

The directory servers that participate in the replication of a given naming context form a directory replication group (DRG). The relationship among the directory servers in a DRG is represented on each node by a special directory entry called a replication agreement. A replication agreement can be either one-way or two-way.

Each copy of a naming context contained within a server is called a replica. A server that sends the updates made to it to other servers is known as a supplier. A server that accepts those changes is called a consumer. A server can be both a supplier and a consumer.

A directory replication group can be single-master, multimaster, or fan-out as described in Table 6-4.

Table 6-4 Types of Directory Replication Groups

Group	Description
Single-master	Has only one supplier replicating changes to one or more consumers. Clients can update only the master node, and can only read data on any of the consumers. This type of group typically uses LDAP.
Multimaster	Enables multiple sites, acting as equals, to manage groups of replicated data. In a multimaster replication environment, each node is both a supplier and a consumer node. Multimaster replication can use LDAP as its transport mechanism. The full DIT is replicated on each node. Replication is always peer-to-peer.
Fan-out	Also called a point-to-point replication group, has a supplier replicating directly to a consumer. That consumer can then replicate to one or more other consumers. Fan-out uses LDAP as its transport mechanism. The replication can be either full or partial. It can be either one-way or two-way.

6.3.5 Example for Single-master, Multimaster, or Fan-out Directory Replication Group Type

Understand about Single-master, Multimaster, or Fan-out Directory Replication group types from examples provided in the following sections.

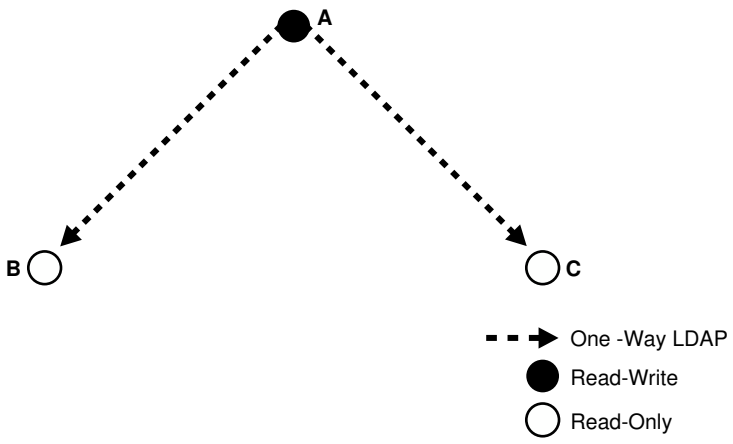
This section has the following topics:

- Single-Master Replication Example
- Multimaster Replication Example
- Fan-out Replication Example

6.3.5.1 Single-Master Replication Example

Figure 6-3 shows a single-master replication environment.

Figure 6-3 Example of Single-Master Replication

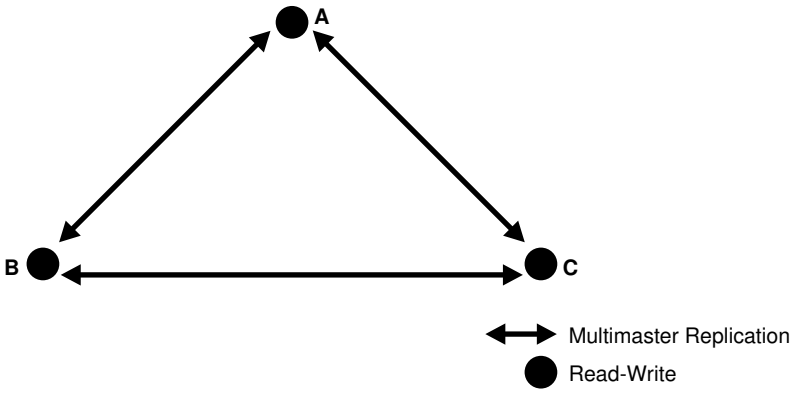


In Figure 6-3, each bullet represents a node of Oracle Internet Directory. Node A is a supplier that replicates consumer nodes B and C. Node A is read/write, and Nodes B and C are read-only. The data transfer protocol is LDAP.

6.3.5.2 Multimaster Replication Example

The example in Figure 6-4 shows three nodes—A, B, and C—that update each other in a multimaster replication group. Replication between nodes is two-way.

Figure 6-4 Example of Multimaster Replication



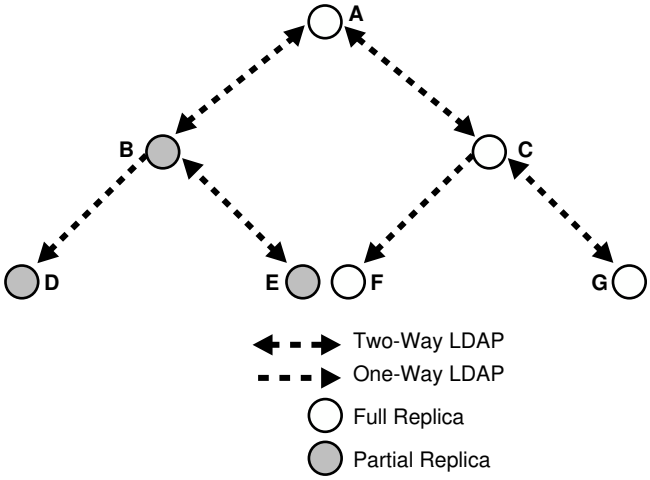
In Figure 6-4, all replication is two-way.

> **Note:**
>
> Multimaster replication is the only replication mechanism supported in Oracle Single Sign-On, as described in the section "Configuring Oracle Single Sign-On for Replication" in the chapter on high availability in the *Oracle Application Server Single Sign-On Administrator's Guide* in the 14c guide.

6.3.5.3 Fan-out Replication Example

Figure 6-5 shows a fan-out replication environment.

Figure 6-5 Example of Fan-Out Replication



In Figure 6-5, supplier A replicates to two consumers, B and C. Consumer node B contains a partial replica of A, whereas consumer node C contains a full replica of A.

Each of these nodes, in turn, serves as a supplier that replicates data to two other consumers: Node B partially replicates to nodes D and E, and node C fully replicates to nodes F and G. Nodes D and F are read-only.

In fan-out replication, nodes transfer data by using LDAP.

6.3.6 Loose Consistency Model in Directory Replication Architecture

Directory replication architecture is based on a loose consistency model: Two replicated nodes in a replication agreement are not guaranteed to be consistent in real time. This increases the overall flexibility and availability of the directory network, because a client can modify data without all interconnected nodes being available.

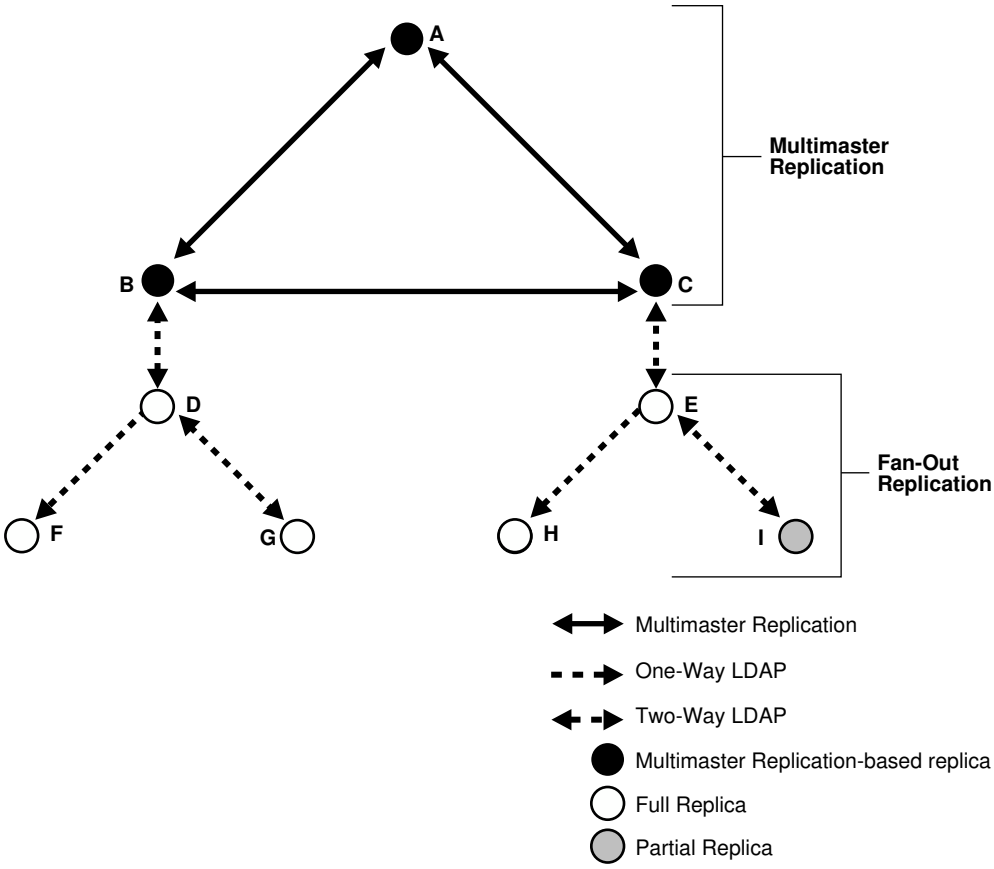
Suppose, for example, that one node is unavailable or heavily loaded. With multimaster replication, the operation can be performed on an alternate node, and all interconnected nodes synchronize in due course.

6.3.7 Multimaster Replication with Fan-Out

Oracle Internet Directory enables any node in a multimaster replication group to also participate in a fan-out replication agreement. Within the multimaster replication agreement, data transfer between the nodes occurs by way of LDAP. Within the fan-out replication agreement, data transfer from supplier to consumer occurs by way of LDAP and can be either one-way or two-way.

Figure 6-6 shows an example of multimaster replication used with fan-out replication.

Figure 6-6 Example of Multimaster Replication with Fan-Out



In the example in Figure 6-6, nodes A, B, and C form a multimaster replication group. They transfer data among them by using LDAP.

Node B supplies changes to Node D, a replica of the entire directory. Node D, in turn, supplies changes to Nodes F and G by using LDAP-based replication. Both Nodes F and G are replicas of the entire directory. Similarly, Node E is a full replica of Node C. Node E, in turn supplies changes to Node H, a replica of the entire directory, and Node I, a partial replica, by using LDAP-based replication. Nodes F and H are read-only.

See LDAP-based replication.

6.4 What Kind of Replication Do You Need?

The type of replication you need depends on the features that are important in your enterprise. This section discusses the types of replication to use to achieve three features: local availability, load balancing, and high availability.

Local Availability

Local availability is important in the following situations:

- You need a local copy of the master data that is specific to the local site. This might be the entire DIT or a partial DIT.

- You need the increased scalability and performance that you can get by distributing the work load among multiple servers.
- You want to reduce the network dependency and network load caused by users from local sites connecting to the master site.

If the local site does not update the data, use one-way replication from master node to local nodes

If the local site does need to update the data and the updates must be replicated back to the master site, use two-way replication between master node and local nodes.

Load Balancing

Load Balancing is important in the following situations:

- You want to increase scalability and performance by distributing work load among multiple servers
- You want to dedicate specific servers to certain applications, such as single sign-on or e-mail applications
- You require network load balancing.

You can use LDAP multi-master replication to replicate the data between two or more nodes.

> **Note:**
>
> When you configure a load balancer for use with replication, use sticky routing and not round-robin routing. Replication is asynchronous in nature, so changes made on one node are not available immediately on the other nodes.

High Availability

High availability is important in the following situations:

- You require a high availability solution with at least one backup server to prevent loss from single server failure
- You need a disaster recovery solution that uses geographically distributed deployments to protect applications from disasters.

Usually, you can use LDAP-based multimaster replication to replicate the data between two or more nodes.

Part II
Basic Administration

This part guides you through common Oracle Internet Directory configuration and maintenance tasks. This part contains these chapters:

- Getting Started With Oracle Internet Directory
- Managing Oracle Internet Directory Instances
- Managing System Configuration Attributes
- Managing IP Addresses in Oracle Internet Directory
- Managing Naming Contexts in Oracle Internet Directory
- Managing Accounts and Passwords in Oracle Internet Directory
- Managing Directory Entries in Oracle Internet Directory
- Managing Dynamic and Static Groups in Oracle Internet Directory
- Performing Bulk Operations
- Managing Collective Attributes
- Managing Alias Entries
- Managing Attribute Uniqueness Constraint Entries
- Managing Knowledge References and Referrals
- Managing Directory Schema
- Configuring Referential Integrity
- Managing Auditing
- Managing Logging
- Monitoring Oracle Internet Directory
- Backing Up and Restoring Oracle Internet Directory

7

Getting Started With Oracle Internet Directory

This chapter describes how to get started with Oracle Internet Directory, including the first tasks you must perform as an Oracle Internet Directory administrator. These tasks include patching your system, performing post installation tasks, and using management interfaces such as Oracle Enterprise Manager Fusion Middleware Control, Oracle Directory Services Manager (ODSM), and command-line utilities.

> **Note:**
>
> Before you perform the tasks in this chapter, Oracle Internet Directory must be installed and configured, as described in the About the Oracle Internet Directory Installation in *Installing and Configuring Oracle Internet Directory*.

This chapter includes the following sections:

- Overview of Postinstallation Tasks and Information
- Overview of Using Fusion Middleware Control to Manage Oracle Internet Directory
- Overview of Oracle Directory Services Manager
- Overview of Managing Oracle Internet Directory Using Command-Line Utilities
- Basic Tasks for Configuring and Managing Oracle Internet Directory

7.1 Overview of Postinstallation Tasks and Information

Perform the following tasks after you complete installation and basic configuration of Oracle Internet Directory.

This section contains the following topics:

- Setting Up the Environment
- Adding Datafiles to the OLTS_CT_STORE and OLTS_ATTRSTORE Tablespaces
- Changing Settings of Windows Services
- Starting and Stopping the Oracle Stack
- Default URLs and Ports
- About Tuning Oracle Internet Directory
- Enabling Anonymous Binds
- Enabling Oracle Internet Directory to run on Privileged Ports
- Verifying Oracle Database Time Zone

7.1.1 Setting Up the Environment

You need to set up the environment first before performing the other tasks.

Set the environment variables described at the beginning of Overview of Managing Oracle Internet Directory Using Command-Line Utilities.

7.1.2 Adding Datafiles to the OLTS_CT_STORE and OLTS_ATTRSTORE Tablespaces

You can skip this step if you have a fresh installation of Oracle Internet Directory 11g Release 1 (11.1.1.6.0) or newer. In that case your Oracle Internet Directory schemas were created by using the Release 1 (11.1.1.6.0) or newer versions of RCU and `config.sh`.

If your schemas were created during installation of a version prior to 11g Release 1 (11.1.1.6.0), you must add datafiles to the OLTS_CT_STORE and OLTS_ATTRSTORE tablespaces if you intend to add more than a million entries to Oracle Internet Directory. Perform this step prior to the `bulkload` or `ldapadd` operation. For details, see Creating Datafiles and Adding Datafiles to a Tablespace in *Oracle Database Administrator's Guide*.

7.1.3 Changing Settings of Windows Services

Change the Startup type of the following Windows services from Automatic to Manual: Oracle Database and TNS Listener. This is necessary to ensure that the services start in the correct order.

7.1.4 Starting and Stopping the Oracle Stack

Understand how to start and stop the components of the Oracle stack in a specific order.

See appendix Starting and Stopping the Oracle Stack for information.

7.1.5 Default URLs and Ports

Get introduced to the default URLs and Ports from the following table.

URL or Port	Default Value
Oracle Directory Services Manager (ODSM)	`http://host:7001/odsm`
Oracle Enterprise Manager Fusion Middleware Control	`http://host:7001/em/`
Oracle WebLogic Server Administrative Console	`http://host:7001/console/`
Oracle Internet Directory LDAP	`3060`
Oracle Internet Directory LDAPS	`3131`

7.1.6 About Tuning Oracle Internet Directory

The default Oracle Internet Directory configuration must be tuned in almost all deployments. You must change the values of the certain configuration attributes, based on your deployment.

See Basic Tuning Considerations, especially the tables Database Parameters and LDAP Server Attributes in *Tuning Performance*.

For more information about tuning, see Oracle Internet Directory Performance Tuning in *Tuning Performance*.. For descriptions of all the attributes, see Managing System Configuration Attributes and Managing Replication Configuration Attributes.

7.1.7 Enabling Anonymous Binds

Anonymous searches, except those on the root DSE, are disabled by default. In some deployment environments, clients might need access to more than the root DSE. If you have such a deployment, set the `orclanonymousbindsflag` attribute to 1.

> **See Also:**
>
> Managing Anonymous Binds

7.1.8 Enabling Oracle Internet Directory to run on Privileged Ports

In many operating systems, only processes running with super user privilege can use port numbers less than 1024. By default, the Installer does not assign privileged ports to Oracle Internet Directory, although you can override the default by explicitly specifying those values via Installer and WLST command inputs.

If you want to change the SSL and non-SSL ports to numbers in the privileged range after installation, proceed as follows:

As a root user, execute the following command:

```
ORACLE_HOME/oidRoot.sh
```

> **Note:**
>
> If you do not have access to super user privileges, have your system administrator execute that script.

Reassign the port numbers in one of the following ways:

- Change the values of `orclnonsslport` and `orclsslport` in the instance-specific configuration entry by using `ldapmodify`, as described in Setting System Configuration Attributes by Using ldapmodify
- Change the SSL Port and Non-SSL Port values on the **General** tab of the **Server Properties** page of Oracle Internet Directory in Oracle Enterprise Manager Fusion Middleware Control, as described in Configuring Server Properties.

Restart Oracle Internet Directory, as described in Restarting the Oracle Internet Directory Server by Using Fusion Middleware Control or Starting Oracle Internet Directory by Using WLST Command.

7.1.9 Verifying Oracle Database Time Zone

To ensure that the Oracle Internet Directory garbage collection logic works correctly, verify the Oracle Database `dbtimezone` parameter.

See Setting Oracle Database Time Zone for Garbage Collection to verify the Oracle Database `dbtimezone` parameter.

7.2 Overview of Using Fusion Middleware Control to Manage Oracle Internet Directory

Oracle Enterprise Manager Fusion Middleware Control is a graphical user interface that provides a comprehensive systems management platform for Oracle Fusion Middleware. Fusion Middleware Control organizes a wide variety of performance data and administrative functions into distinct, Web-based home pages for the domain, Oracle instances, middleware system components, and applications.

This section contains the following topics:

- Managing Oracle Internet Directory Using Fusion Middleware Control
- Oracle Internet Directory Menu

7.2.1 Managing Oracle Internet Directory Using Fusion Middleware Control

Understand how to manage Oracle Internet Directory using Fusion Middleware Control. Oracle Internet Directory is a target type in Oracle Enterprise Manager Fusion Middleware Control.

> **Note:**
>
> - If you selected **Configure Without a Domain** when prompted for a domain while installing Oracle Internet Directory, Oracle Enterprise Manager Fusion Middleware Control will not be available.
> - Oracle Enterprise Manager Fusion Middleware Control manages Oracle Internet Directory through its SSL port. The Oracle Internet Directory SSL port must be configured for no authentication or server authentication. In addition, the ciphers configured must include one or more of the Diffie-Hellman no-auth ciphers:
>   - `SSL_DH_anon_WITH_RC4_128_MD5`
>   - `SSL_DH_anon_WITH_DES_CBC_SHA`
>
>   Oracle Enterprise Manager Fusion Middleware Control manages Oracle Internet Directory through its SSL port. Set `orclsslenable` to `1` or `2` if you use WLST or Oracle Enterprise Manager Fusion Middleware Control to configure the server. See About SSL Authentication Modes.
>
>   If the Oracle Internet Directory SSL port is configured incorrectly, or if the appropriate ciphers are not configured, you will not be able to change Oracle Internet Directory parameters by using WLST or Oracle Enterprise Manager Fusion Middleware Control. See About SSL Authentication Modes.
> - For information about supported browsers for Fusion Middleware Control and Oracle Directory Services Manager, refer to System Requirements and Supported Platforms for Oracle Fusion Middleware 11gR1, which is linked from: `http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html`

To use the interface to Oracle Internet Directory:

1. Connect to Fusion Middleware Control.

   The URL is of the form:

   `http://host:port/em`

2. In the left panel topology tree, expand the domain, then Fusion Middleware, then Identity and Access. Alternatively, from the domain home page, expand Fusion Middleware, then Identity and Access. Instances of Oracle Internet Directory are listed in both places. To view the full name of a component instance, move the mouse over the instance name.
3. Select the Oracle Internet Directory component you want to manage.
4. Use the Oracle Internet Directory menu to select tasks.

7.2.2 Oracle Internet Directory Menu

You can use the Oracle Internet Directory menu to navigate to other Fusion Middleware Control pages for Oracle Internet Directory, navigate to Oracle Directory Services Manager

pages for Oracle Internet Directory, and perform other tasks, as described in the following table.

Table 7-1 Using the Oracle Internet Directory Menu

Task	Select
Return to Home page	**Home**
View a performance summary	**Monitoring**, then **Performance**
Start, stop, or restart the Oracle Internet Directory component	**Control**, then **Start Up**, **Shut Down**, or **Restart**, respectively.
View Oracle Internet Directory logs	**Logs**, then **View Log Messages**
View non-SSL and SSL port information.	**Port Usage**
Manage properties that are specific to this Oracle Internet Directory component	**Administration**, then **Server Properties**
Manage properties that are shared by all Oracle Internet Directory components that are connected to the same Oracle Database	**Administration**, then **Shared Properties**
Manage Oracle Internet Directory entries by using Oracle Directory Services Manager	**Directory Services Manager**, then **Data Browser**
Manage the Oracle Internet Directory schema by using Oracle Directory Services Manager	**Directory Services Manager**, then **Schema**
Manage Oracle Internet Directory security by using Oracle Directory Services Manager	**Directory Services Manager**, then **Security**
Manage Oracle Internet Directory advanced features by using Oracle Directory Services Manager	**Directory Services Manager**, then **Advanced**
Configure auditing for Oracle Internet Directory	**Security**, then **Audit Policy Settings**
View target name, software version, Oracle home, Oracle instance, Oracle Enterprise Manager Fusion Middleware Control agent, and host	**General Information**.

7.3 Overview of Oracle Directory Services Manager

Oracle Directory Services Manager is a web-based interface for managing instances of Oracle Internet Directory and Oracle Virtual Directory. It is a replacement for Oracle Directory Manager, which is now deprecated.

This section contains the following topics:

- Understanding Oracle Directory Services Manager
- Configuring ODSM for SSO Integration
- Configuring the SSO Server for ODSM Integration
- About Configuring the Oracle HTTP Server for ODSM-SSO Integration
- Invoking Oracle Directory Services Manager
- Overview of Connecting to the Server from Oracle Directory Services Manager
- Configuring Oracle Directory Services Manager Session Timeout
- Configuring Oracle HTTP Server to Support Oracle Directory Services Manager in an Oracle WebLogic Server Cluster

See Managing Oracle Directory Services Manager's Java Key Store.

7.3.1 Understanding Oracle Directory Services Manager

Oracle Directory Services Manager is a web-based interface for managing instances of Oracle Internet Directory and Oracle Virtual Directory. It is a replacement for Oracle Directory Manager, which is now deprecated. Oracle Directory Services Manager enables you to configure the structure of the directory, define objects in the directory, add and configure users, groups, and other entries. ODSM is the interface you use to manage entries, schema, security, and other directory features.

You can also use ODSM to manage system configuration attributes, which can be useful if Fusion Middleware Control is not available or if you must modify an attribute that has no Fusion Middleware Control interface. See Managing System Configuration Attributes by Using ODSM Data Browser and Managing Entries by Using Oracle Directory Services Manager.

This section includes the following topics:

- About the JAWS Screen Reader with Oracle Directory Services Manager
- Non-Super User Access to Oracle Directory Services Manager
- Single Sign-On Integration with Oracle Directory Services Manager

7.3.1.1 About the JAWS Screen Reader with Oracle Directory Services Manager

When you use JAWS with ODSM, whenever a new window pops up, JAWS reads "popup." To read the entire page, enter the keystrokes Insert+b.

7.3.1.2 Non-Super User Access to Oracle Directory Services Manager

Oracle Directory Services Manager allows you to connect to Oracle Internet Directory as any user with a valid DN and password in the directory. If you connect as the super user, `cn=orcladmin`, or as a user who is a member of `cn=DirectoryAdminGroup,cn=oracle internet directory`, you can access all the tabs in the interface. If you log in as any other user, you can access only the Home, Schema, and Data Browser tabs.

7.3.1.3 Single Sign-On Integration with Oracle Directory Services Manager

You can configure Oracle Directory Services Manager to use Single Sign-On (SSO). When configured with SSO, Oracle Directory Services Manager allows a user who has been authenticated by the SSO server to connect to an SSO-enabled directory without logging in, provided that user has privileges to manage the directory.

Oracle Directory Services Manager maintains a list of Oracle Internet Directory servers that SSO-authenticated users can manage. To validate whether an SSO-authenticated user has the required privileges to manage Oracle Internet Directory, Oracle Directory Services Manager maps the SSO-authenticated user to a DN in the Oracle Internet Directory server.

Oracle Directory Services Manager uses proxy authentication to connect to the directory. The proxy user's DN and password are stored in a secure storage framework called the Credential Store Framework (CSF).

To map an SSO-authenticated user, Oracle Directory Services Manager authenticates to the Oracle Internet Directory server using the credentials of a user with proxy privileges. Oracle Directory Services Manager then tries to map the SSO-authenticated user's unique identifier to the Oracle Internet Directory user's unique identifier.

The WLS Administrator configures the proxy user's credentials, unique identifier attribute, and the base DN under which Oracle Directory Services Manager searches for the user, which are stored in the CSF. If Oracle Directory Services Manager gets a valid DN, it maps the SSO-authenticated user to that DN. When the SSO-authenticated user is mapped to a valid DN, Oracle Directory Services Manager uses proxy authentication to connect to the Oracle Internet Directory server with the SSO-authenticated user's mapped DN.

To configure SSO integration, see Configuring ODSM for SSO Integration, Configuring the SSO Server for ODSM Integration, and About Configuring the Oracle HTTP Server for ODSM-SSO Integration.

7.3.2 Configuring ODSM for SSO Integration

To configure ODSM-SSO integration, use the ODSM Proxy Bind Configuration Screen, at `http://host:port/odsm-config`. Log in as the WebLogic administrator.

On this screen, you provide Oracle Directory Services Manager with the set of directory servers that SSO users can manage. This screen lists the Single Sign-On accessible directories.

Use the **View** list to modify the number and order of the columns. To remove an existing directory, click **Remove**.

To modify an existing directory, click **Modify**.

To add a new Single Sign-On accessible directory, click **Add**.

When you click **Modify** or **Add**, the **Directory Details** screen appears. Proceed as follows:

1. Select **Non-SSL** or **SSL** from the **Port Type** list.
2. Select **OID** or **OVD** from the **Directory Type** list.
3. Provide the following information:
   - **Host** and **Port** of the directory.
   - **Proxy User's DN** and **Password**: The DN and password that Oracle Directory Services Manager uses for proxy authentication.
   - **User Container DN**: The DN under which user entries are located in the directory.
   - **User Lookup Attribute**: A unique attribute for looking up a user's DN in the directory. For example, if the SSO server sends the user's mail ID to Oracle Directory Services Manager as the user's unique identifier, you can configure `mail` as the user look-up attribute.
4. Click **Validate** to verify your directory connection details.

   Oracle Directory Services Manager authenticates to the directory server with the credentials provided.
5. Click **Apply** to apply your selections.

   Click **Revert** to abandon your selections.
6. Specify the SSO server's Logout URL in the SSO Logout URL text box.

   For example, http://*myoamhost.mycompany.com:*14100/oam/server/logout is the default Logout URL for the Oracle Access Manager 11g server. If you only configure this field, Oracle Directory Services Manager displays the Login link at the top right corner of the Oracle Directory Services Manager page.

7.3.3 Configuring the SSO Server for ODSM Integration

To make SSO-ODSM integration work correctly and to improve performance, you must configure specific ODSM URLs as protected, unprotected, or excluded.

The ODSM home page must be an unprotected URL. That is, all users must be able to access the ODSM home page, including those who have not gone through the SSO authentication process.

The `/odsm/odsm-sso.jsp` URL must be protected by the SSO server. When a user clicks the **Login** link appearing on the top right corner of the home page, ODSM redirects the user to `/odsm/odsm-sso.jsp`. The SSO server challenges the user for a username and password, if the user is not already authenticated. Upon successful authentication, the user is directed back to the ODSM home page.

Configure the ODSM URLs as follows:

- Protected: `/odsm/odsm-sso.jsp`
- Unprotected: `/odsm/faces/odsm.jspx`
- Excluded: `/odsm/.../`

Setting the CSS, JavaScript, and graphics (`/odsm/.../`) files to excluded prevents these files from being validated by Oracle Access Manager, which can improve the performance of your deployment.

You can use either Oracle Access Manager 14*c* or Oracle Access Manager 14*c* as your SSO provider.

You must configure an Oracle Access Manager server to send the SSO-authenticated user's unique identifier through an HTTP header to Oracle Directory Services Manager. Oracle Directory Services Manager looks for the `OAM_REMOTE_USER HTTP` header. The Oracle Access Manager server sets the `OAM_REMOTE_USER` header by default. If this header is not available, Oracle Directory Services Manager looks for the `odsm-sso-user-unique-id` HTTP header. If Oracle Directory Services Manager cannot find any of these headers, Oracle Directory Services Manager SSO integration will not work.

In addition to sending the user's unique identifier through HTTP header, you can optionally configure Oracle Access Manager to send following HTTP headers:

- Configure the `odsm-sso-user-firstname` HTTP header to send the user's first name.
- Configure the `odsm-sso-user-lastname` HTTP header to send the user's last name.

If these headers are available, Oracle Directory Services Manager displays the user's first name and last name in the "Logged in as" section located in the top right corner of Oracle Directory Services Manager. If the first name or the last name is not available, Oracle Directory Services Manager displays the user's unique identifier in the "Logged in as" section.

To configure Oracle Access Manager 14*c*, see "Deploying the OAM 11g SSO Solution" chapter in *Oracle Fusion Middleware Application Security Guide*.

7.3.4 About Configuring the Oracle HTTP Server for ODSM-SSO Integration

If you are using Oracle HTTP Server to host the SSO server's WebGate agent and as a front end to the WebLogic server hosting ODSM, you must configure Oracle HTTP Server's `mod_wl_ohs` module to forward all requests starting with `/odsm` to the WebLogic server hosting

ODSM. The `mod_wl_ohs` module allows requests to be proxied from Oracle HTTP Server to Oracle WebLogic Server.

To configure `mod_wl_ohs`, see mod_wl_ohs in *Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server*.

7.3.5 Invoking Oracle Directory Services Manager

You can invoke Oracle Directory Services Manager directly or from Oracle Enterprise Manager Fusion Middleware Control.

> **Note:**
>
> - If you selected **Configure Without a Domain** when prompted for a domain while installing Oracle Internet Directory, Oracle Directory Services Manager will not be available.
> - For information about supported browsers for Fusion Middleware Control and Oracle Directory Services Manager, refer to System Requirements and Supported Platforms for Oracle Fusion Middleware, which is linked from: `http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html`

- To invoke Oracle Directory Services Manager directly, enter the following URL into your browser's address field:

  `http://host:port/odsm`

  In the URL to access Oracle Directory Services Manager, *host* is the name of the admin server where Oracle Directory Services Manager is running. *port* is the admin server port number from the WebLogic server. You can determine the exact port number by examining the $*Fusion_Middleware_Home*/*Oracle_Identity_Management_domain*/servers/wls_ods/data/nodemanager/wls_ods1.url file, where *Fusion_Middleware_Home* represents the root directory where Fusion Middleware is installed.
- To invoke Oracle Directory Services Manager from Fusion Middleware Control, select **Directory Services Manager** from the Oracle Internet Directory menu in the Oracle Internet Directory target, then **Data Browser**, **Schema**, **Security**, or **Advanced**. (You can connect from the Oracle Virtual Directory menu in a similar manner.)

  A new browser window, containing the ODSM Welcome screen, pops up. Connect to the server as described in the next section.

See Troubleshooting Oracle Directory Services Manager.

7.3.6 Overview of Connecting to the Server from Oracle Directory Services Manager

When the ODSM Welcome screen appears, you can connect to either an Oracle Internet Directory server or a Oracle Virtual Directory server.

This section contains the following topics:

- Logging into the Directory Server from Oracle Directory Services Manager

- Understanding Logging Into the Directory Server from Oracle Directory Services Manager Using SSL

> **Note:**
>
> - After you have logged into ODSM, you can connect to multiple directory instances from the same browser window.
> - Avoid using multiple windows of the same browser program to connect to different directories at the same time. Doing so can cause a `Target unreachable` error.
> - You can log in to the same ODSM instance from different browser programs, such as Internet Explorer and Firefox, and connect each to a different directory instance.
> - If you change the browser language setting, you must update the session in order to use the new setting. To update the session, either reenter the ODSM URL in the URL field and press **Enter** or quit and restart the browser.

7.3.6.1 Logging into the Directory Server from Oracle Directory Services Manager

You log in to a directory server's non-SSL port from Oracle Directory Services Manager as follows:

1. Click the small arrow to the right of the label **Click to connect to a directory**. It opens a dialog box containing the following sections:
   - Live Connections–current connections that you can return to.
   - Disconnected Connections–a list of directory servers you have connected to and then disconnected from. Oracle Directory Services Manager saves information about connections that you've used previously and lists them, by optional Name or by server, so that you can select them again.

     > **Note:**
     >
     > If the connection information changes, for example, if the server's port number changes, Oracle Directory Services Manager's connection information becomes incorrect and the connection fails. You cannot edit connection information, so you must delete the connection from the list and create a new connection.

   - New Connections–used to initiate a new connection

   If you are SSO-authenticated, you might see an additional section, described in Connecting to an SSO-Enabled Directory as an SSO-Authenticated User.
2. To reconnect to a live connection, click it.

   To select a disconnected connection, click the entry. You see a short version of the Login Dialog with most fields filled in. To remove a selection from the list, select it and then select Delete.

   To initiate a connection to a new directory server, click **Create a New Connection** or type Ctrl+N. The New Connection Dialog appears.

3. Select **OID** or **OVD**.
4. Optionally, enter an alias name to identify this entry on the Disconnected Connections list.
5. Enter the server and non-SSL port for the Oracle Internet Directory or Oracle Virtual Directory instance you want to manage.
6. Deselect **SSL Enabled**.
7. Enter the user (usually `cn=orcladmin`) and password.
8. Select the Start Page you want to go to after logging in.
9. Click **Connect**.

After you have logged in to an Oracle Internet Directory or Oracle Virtual Directory server, you can use the navigation tabs to select other pages.

The Oracle Directory Services Manager home pages for Oracle Internet Directory and Oracle Virtual Directory list version information about Oracle Directory Services Manager itself, as well as the directory and database. It also lists directly statistics.

> **See Also:**
>
> Troubleshooting Oracle Directory Services Manager.

7.3.6.2 Understanding Logging Into the Directory Server from Oracle Directory Services Manager Using SSL

If you are unfamiliar with SSL authentication modes, see About SSL Authentication Modes.

When you log in to the server's SSL port, you follow the procedure in Logging into the Directory Server from Oracle Directory Services Manager, except that you specify the SSL port in Step 5 and do not deselect **SSL Enabled** in Step 6. After you click **Connect** in Step 9, you might be presented with a certificate, depending on the type of SSL authentication.

This section contains the following topics:

- SSL No Authentication
- SSL Server Only Authentication
- SSL Client and Server Authentication

7.3.6.2.1 SSL No Authentication

If the directory server is using SSL No Authentication mode (the default), you are not presented with a certificate. SSL No Authentication provides data confidentiality and integrity only but no authentication using X509 certificates.

7.3.6.2.2 SSL Server Only Authentication

If the directory server is using SSL Server Authentication Only Mode, when you click connect in Step 9 in Logging into the Directory Server from Oracle Directory Services Manager, you are presented with the server's certificate. After manually verifying the authenticity of the server certificate, you can accept the certificate permanently, accept the certificate for the current session only, or reject the certificate. If you accept the certificate permanently, the certificate is stored in its Java Key Store (JKS). From then on, you are not prompted to accept the

certificate when you connect to that server. If you accept the certificate only for the current session, you are prompted to accept or reject the certificate every time you connect to the server. If you reject the certificate, ODSM closes the connection to the server.

> **See Also:**
>
> Introduction to Managing ODSM's Java Key Store.

7.3.6.2.3 SSL Client and Server Authentication

If the server is using SSL Client and Server Authentication Mode, when you click **Connect** in Step 9 in Logging into the Directory Server from Oracle Directory Services Manager, you are presented with a certificate. Follow the instructions in SSL Server Only Authentication.

After ODSM accepts the server's certificate, ODSM sends its own certificate to the server for authentication. The server accepts ODSM's certificate if that certificate is present in its trusted list of certificates.

If the DN of ODSM's certificate is present in the server, you do not need to provide the username and password in the connection dialog.

If the DN of ODSM's certificate is not present in the server, you must provide the user name and password.

ODSM's certificate is a self-signed certificate. You must use the `keytool` command to assign a CA signed certificate to ODSM. See Managing Oracle Directory Services Manager's Java Key Store.

7.3.6.3 Connecting to an SSO-Enabled Directory as an SSO-Authenticated User

If you have already been authenticated by the single sign-on server, ODSM allows you to connect to SSO-enabled directories without logging in, provided you have an entry in that directory. When you access the ODSM Welcome page, if you have an entry in only one SSO-enabled directory, ODSM connects you to it. If you have entries in more than one SSO-enabled directory ODSM allows you to select directory you want to connect to, as follows.

Click the small arrow to the right of the label **Click to connect to a directory**. In this case, the dialog box contains an extra section, listing SSO-enabled directories you are authorized to connect to. Select the directory you want. ODSM connects you without requesting a username or password.

If the port you connected to is an SSL port, you still must perform the appropriate steps in SSL No Authentication, SSL Server Only Authentication, or SSL Client and Server Authentication.

7.3.7 Configuring Oracle Directory Services Manager Session Timeout

Beginning with Oracle Internet Directory 11g Release 1 (11.1.1.9.0), the default session timeout for Oracle Directory Services Manager (ODSM) is 5 minutes (300 seconds). You can set the ODSM session timeout to a different value using the WebLogic Server Administration Console. (In earlier releases, you set the timeout by editing the `<session-config>` element in the `web.xml` deployment descriptor.)

To configure the ODSM session timeout:

1. Log in to the WebLogic Server Administration Console.

2. In the Change Center, click **Lock and Edit**.
3. In the left pane, click **Deployments**.
4. Under Deployments, expand **odsm**.
5. Under Modules, click **/odsm**.
6. Select the **Configuration** tab.
7. Set the **Session Timeout (in seconds)** to the required value. For example: 600 seconds.
8. Click **Save**.
9. Under Save Deployment Plan Assistant, click **OK** to save the change to the `Plan.xml` file.
10. In the Change Center, click **Activate Changes**.

To test your change, access ODSM, log in to a directory server, and leave the session idle for the time you specified for the session timeout (for example, 600 seconds). The session times out after the specified time, and ODSM displays the session timeout popup message.

7.3.8 Configuring Oracle HTTP Server to Support Oracle Directory Services Manager in an Oracle WebLogic Server Cluster

Perform the following steps to configure Oracle HTTP Server to route Oracle Directory Services Manager requests to multiple Oracle WebLogic Servers in a clustered Oracle WebLogic Server environment

To configure Oracle HTTP Server:

1. Create a backup copy of the Oracle HTTP Server's `httpd.conf` file. The backup copy provides a source to revert to if you encounter problems after performing this procedure.
2. Add the following text to the end of the Oracle HTTP Server's `httpd.conf` file and replace the variable placeholder values with the host names and managed server port numbers specific to your environment. Be sure to use the `<Location /odsm/ >` as the first line in the entry. Using `<Location /odsm/faces >` or `<Location /odsm/faces/odsm.jspx >` can distort the appearance of the Oracle Directory Services Manager interface.

   ```
   <Location /odsm/ >
   SetHandler weblogic-handler
   WebLogicCluster host-name-1:managed-server-port,host-name_2:managed_server_port
   </Location>
   ```

3. Stop, then start the Oracle HTTP Server to activate the configuration change.

> **Note:**
>
> Oracle Directory Services Manager loses its connection and displays a session time-out message if the Oracle WebLogic Server in the cluster that it is connected to fails. Oracle Directory Services Manager requests are routed to the secondary Oracle WebLogic Server in the cluster that you identified in the httpd.conf file after you log back in to Oracle Directory Services Manager.

7.4 Overview of Managing Oracle Internet Directory Using Command-Line Utilities

Command-Line utilities can be used to manage Oracle Internet Directory. You need to set specific environmental variables to use most Oracle Internet Directory command-line utilities.

This section contains the following topics:

- About Setting Environmental Variables to Use Oracle Internet Directory Command-Line Utilities
- About Standard LDAP Utilities
- Bulk Tools
- About WLST

7.4.1 About Setting Environmental Variables to Use Oracle Internet Directory Command-Line Utilities

To use most Oracle Internet Directory command-line utilities and Database client utilities like sqlplus, you must set the following environmental variables.

To set the environmental variables:

- `ORACLE_HOME` - The location of non-writable files in your Oracle Identity Management installation.
- `DOMAIN_HOME` - The location of writable files in your Oracle Identity Management installation.
- `TNS_ADMIN` - The directory where the database connect string is defined in the tnsnames.ora file. By default it is the `$DOMAIN_HOME/config/fmwconfig/components/OID/`*`componentName`*`/config/` directory. The database connect alias as defined in `tnsnames.ora` is `OIDDB` by default.
- `NLS_LANG` (*`APPROPRIATE_LANGUAGE`*`.AL32UTF8`) - The default language set at installation is `AMERICAN_AMERICA`.
- `PATH` - The following directory locations should be added to your `PATH`:

  `$`*`ORACLE_HOME`*`/bin`

  `$`*`ORACLE_HOME`*`/ldap/bin`

  `$`*`DOMAIN_HOME`*`/bin`

Many of the activities that you can perform at the command line can also be performed in Oracle Enterprise Manager Fusion Middleware Control or Oracle Directory Services Manager. A few functions are only available from the command line.

7.4.2 About Standard LDAP Utilities

Oracle Internet Directory supports the standard LDAP command-line utilities `ldapadd`, `ldapaddmt`, `ldapbind`, `ldapcompare`, `ldapdelete`, `ldapmoddn`, `ldapmodify`, `ldapmodifymt`, and `ldapsearch`.

For example:

```
ldapbind -D "cn=orcladmin" -q -h "myserver.example.com" -p 3060

ldapsearch -b "cn=subschemasubentry" -s base "objectclass=*" -p 3060 \
     -D "cn=orcladmin" -q
```

This book contains many examples of LDAP tool use.

> **See Also:**
>
> - Managing Entries by Using LDAP Command-Line Tools.
> - Oracle Internet Directory Data Management Tools in *Reference for Oracle Identity Management* for a detailed description of each tool.

For security reasons, avoid supplying a password on the command line whenever possible. A password typed on the command line is visible on your screen and might appear in log files or in the output from the `ps` command.

When you supply a password at a prompt, it is not visible on the screen, in `ps` output, or in log files. Use the `-Q` and `-q` options, respectively, instead of the `-P` *`password`* and `-w` *`password`* options. If there is no wallet password and you are using the `-Q` option, when prompted for the password, hit Enter.

The `LDAP` tools have been modified to disable the options `-w` *`password`* and `-P` *`password`* when the environment variable `LDAP_PASSWORD_PROMPTONLY` is set to `TRUE` or `1`. Use this feature whenever possible.

See Using Passwords with Command-Line Tools in *Reference for Oracle Identity Management*.

7.4.3 Bulk Tools

Oracle Internet Directory provides several tools to help you manage large numbers of entries.

See Performing Bulk Operations

> **See Also:**
>
> Oracle Internet Directory Data Management Tools in *Reference for Oracle Identity Management* for a detailed description of each tool.

7.4.4 About WLST

The Oracle WebLogic Scripting Tool (WLST) is a Jython-based command-line scripting environment that you can use to manage and monitor WebLogic Server domains. To use it to manage and monitor Oracle Internet Directory, you must navigate to the custom MBean tree where Oracle Internet Directory is located. Then you can list, get values, and change values of the managed beans (MBeans) that represent Oracle Internet Directory resources.

> **See Also:**
>
> Managing System Configuration Attributes by Using WLST

> **Note:**
>
> WLST manages Oracle Internet Directory through its SSL port. Set `orclsslenable` to `1` or `2` if you use WLST or Oracle Enterprise Manager Fusion Middleware Control to configure the server. See About SSL Authentication Modes.

7.5 Basic Tasks for Configuring and Managing Oracle Internet Directory

Learn about the steps that you must take to configure and manage a basic Oracle Internet Directory environment from the following table.

Table 7-2 Basic Tasks for Configuring and Managing Oracle internet Directory

Task	Reference
Start and stop the LDAP server	See Managing Oracle Internet Directory Instances
Manage system configuration attributes	See Managing System Configuration Attributes
Manage directory entries	See Managing Directory Entries in Oracle Internet Directory
Manage directory schema	See Managing Directory Schema
Configure auditing	See Managing Auditing
Manage log files	See Managing Logging
Configure SSL	See Configuring Secure Sockets Layer (SSL)
Configure password policies	See Managing Password Policies
Configure access control	See Managing Directory Access Control
Get sizing and tuning recommendations for Oracle Internet Directory deployments	See Obtaining Recommendations by Using the Tuning and Sizing Wizard in *Tuning Performance*.
Set up replication	See Setting Up Replication
Modifying an existing replication setup	See Managing and Monitoring Replication

This guide describes other tasks that you might need to perform, depending on your Oracle Fusion Middleware environment.

8

Managing Oracle Internet Directory Instances

This chapter describes Oracle Internet Directory server instances and how to create and manage these instances using Oracle Enterprise Manager Fusion Middleware Control and the `WLST` and OIDCTL utilities.
This chapter includes the following sections:

- Overview of Managing Oracle Internet Directory Instances
- Overview of Oracle Internet Directory Components Management by Using Fusion Middleware Control
- Managing Oracle Internet Directory Components by Using WLST Commands
- Starting an Instance of the Replication Server by Using OIDCTL

> **See Also:**
>
> Managing Oracle Internet Directory Instances by Using OIDCTL

8.1 Overview of Managing Oracle Internet Directory Instances

Understand the process of managing Oracle Internet Directory Instances.

This section contains the following topics:

- About the Instance-Specific Configuration Entry
- About the First Oracle Internet Directory Instance Creation
- Creating Additional Oracle Internet Directory Instances

8.1.1 About the Instance-Specific Configuration Entry

Understand about the instance-specific configuration entry.

Since 11g Release 1 (11.1.1.0.0), configuration information for an Oracle Internet Directory instance resides in an instance-specific configuration entry, which has a DN of the form:

`cn=`*`componentname`*`,cn=osdldapd,cn=subconfigsubentry`

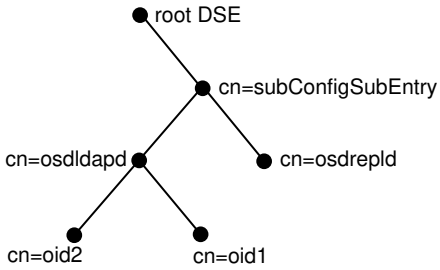
where *`componentname`* is the name of a Oracle Fusion Middleware system component of `Type=OID`, such as `oid1` or `oid2`.

You do not manually create an instance-specific configuration entry. Instead, you create a Oracle Fusion Middleware system component of `Type=OID`, which automatically generates an instance-specific configuration entry named `oid1`.

Figure 8-1 shows the configuration entries for two Oracle Internet Directory components in the DIT. The DNs for the instance-specific configuration entries are:

```
cn=oid1,cn=osdldapd,cn=subconfigsubentry
cn=oid2,cn=osdldapd,cn=subconfigsubentry
```

Figure 8-1 DIT Showing Two Instance-Specific Configuration Entries



The attributes in the instance-specific configuration specify information such as hostname, ports, events to be audited, number of child processes, and security configuration. For a complete list, see Attributes of the Instance-Specific Configuration Entry.

8.1.2 About the First Oracle Internet Directory Instance Creation

Understand when and how the first Oracle Internet Directory Instance gets created.

When you install Oracle Internet Directory on a host computer, a default instance-specific configuration entry named `oid1` is created for the OID component, as follows:

```
cn=oid1,cn=osdldapd,cn=subconfigsubentry
```

The default `oid1` configuration entry is created in collocated mode using the following scenarios:

- Run the installer to layout the binaries
- Run `rcu` to setup Oracle Internet Directory database
- Run `config.sh` to create the Weblogic domain for Oracle Internet Directory
- Start Weblogic admin server and node manager
- Run `oid_setup()` WLST command to create default `oid1` component instance

The Oracle Internet Directory component contains an OIDMON process and an Oracle Internet Directory instance (`inst=1`). The Oracle Internet Directory instance consists of a dispatcher process and one or more OIDLDAPD processes.

Beginning with Oracle Internet Directory 11g Release 1 (11.1.1.7.0), the OIDLDAPD process is separated as the OIDDISPD (dispatcher) process and the OIDLDAPD (server) process. On UNIX and Linux systems, however, the `ps -ef` command will continue to show both of these processes as OIDLDAPD at runtime.

In addition, the configuration step for Oracle Internet Directory creates some file system directories under Weblogic *DOMAIN_HOME* directory. Some of the pathnames it creates are specific to the component name. For example, the pathnames under your Oracle instance on UNIX or Linux include:

```
$DOMAIN_HOME/config/fmwconfig/components/OID/config/componentName
$DOMAIN_HOME/servers/OID/logs/componentName
```

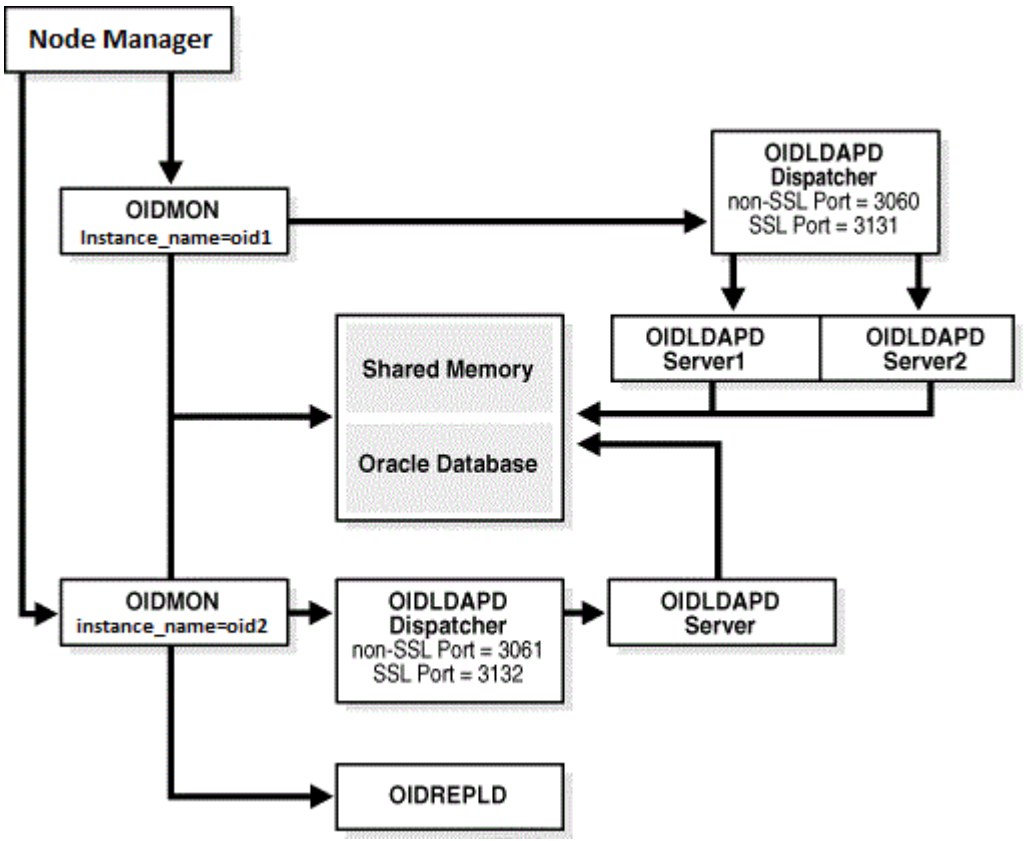
> **Note:**
>
> Oracle Internet Directory is frequently configured in a cluster where instances on different hosts are all connected to the same Oracle Database.
>
> `oid_CreateInstance()` WLST command detects that the other `OID` components are using the same Oracle Database and increments the component name for the new component by 1. That is, successive installations in the cluster will have the component names `oid2`, `oid3`, and so forth.

8.1.3 Creating Additional Oracle Internet Directory Instances

The recommended way to add another Oracle Internet Directory instance is to add an additional system component of `Type=OID` in the Oracle instance.

To do this, use WLST `createInstance` command, specifying the name of the instance, host and the port on which OID server should be running. This new Oracle Internet Directory instance consists of an OIDMON process, an OIDLDAPD dispatcher process, and one or more OIDLDAPD server processes. For example, see `instance_name=oid2` at the bottom of Figure 8-2.

Figure 8-2 Oracle Internet Directory Oracle Internet Directory Process Control Architecture



Use WLST command, `oid_createInstance` , to create a new instance-specific configuration entry in the DIT. If the new component name is `oid2`, the new entry looks like this:

```
cn=oid2,cn=osdldapd,cn=subconfigsubentry
```

Change the values of attributes in this entry to customize the instance.

The `WLST` command also creates additional pathnames in the file system under the *`DOMAIN_HOME`* directory. If the new instance name is `oid2`, the path names include:

```
$DOMAIN_HOME/config/fmwconfig/components/OID/config/oid2
$DOMAIN_HOME/tools/OID/logs
```

You can use WLST commands to manage the components `oid1` and `oid2` individually.

> **Note:**
>
> You can use `oidctl` to create an instance if you are running Oracle Internet Directory as a standalone server, not part of a WebLogic domain. When you create an instance with `oidctl`, you must use `oidctl` to stop and start the instance. An Oracle Internet Directory instance created with `oidctl` cannot be registered with a WebLogic server, so you cannot use Oracle Enterprise Manager Fusion Middleware Control to manage the instance. See Managing Oracle Internet Directory Instances by Using OIDCTL .

> **See Also:**
>
> - Understanding Process Control of Oracle Internet Directory Components for information about Oracle Internet Directory processes.

8.1.4 Registering an Oracle Instance or Component with the WebLogic Server

If you want to manage an Oracle Internet Directory component with Oracle Enterprise Manager Fusion Middleware Control, you must register the component and the Oracle instance that contains it with a WebLogic domain. You can register an Oracle instance with a WebLogic domain during installation or Oracle instance creation, but you are not required to do so.

If the Oracle instance is already registered, and you are adding a new Oracle Internet Directory system component to the Oracle instance, the Node Manager automatically registers the component as part of that Oracle instance.

> **See Also:**
>
> Understanding WebLogic Server Domain.

8.2 Overview of Oracle Internet Directory Components Management by Using Fusion Middleware Control

You can view, stop, and start Oracle Internet Directory components by using Oracle Enterprise Manager Fusion Middleware Control.

This section contains the following topics:

- Viewing Active Server Information by Using Fusion Middleware Control
- Starting the Oracle Internet Directory Server by Using Fusion Middleware Control
- Stopping the Oracle Internet Directory Server by Using Fusion Middleware Control
- Restarting the Oracle Internet Directory Server by Using Fusion Middleware Control

8.2.1 Viewing Active Server Information by Using Fusion Middleware Control

You can view information about any Oracle Internet Directory component—including type, debug level, host name, and configuration parameters— using Oracle Enterprise Manager Fusion Middleware Control.

Follow the steps below:

1. Connect to Oracle Enterprise Manager Fusion Middleware Control as described in Overview of Using Fusion Middleware Control to Manage Oracle Internet Directory .
2. The Domain Home Page displays the status of components, including Oracle Internet Directory.
3. Select the Oracle Internet Directory component you want to view.
4. View the status information on the Oracle Internet Directory Home page.

8.2.2 Starting the Oracle Internet Directory Server by Using Fusion Middleware Control

You can start the Oracle Internet Directory Server using Fusion Middleware Control.

Start the Oracle Internet Directory server as follows:

1. Go to the Oracle Internet Directory home page in Oracle Enterprise Manager Fusion Middleware Control.
2. From the Oracle Internet Directory menu, select **Control**, then **Start Up**.
3. When the confirmation dialog appears, click **OK**.

If Fusion Middleware Control cannot start the server, an error dialog appears.

8.2.3 Stopping the Oracle Internet Directory Server by Using Fusion Middleware Control

You can stop the Oracle Internet Directory Server using Fusion Middleware Control.

Stop the Oracle Internet Directory server as follows:

1. Go to the Oracle Internet Directory home page in Oracle Enterprise Manager Fusion Middleware Control.
2. From the Oracle Internet Directory menu, select **Control**, then **Shut Down**.
3. When the confirmation dialog appears, click **OK**.

If Fusion Middleware Control cannot stop the server, an error dialog appears.

8.2.4 Restarting the Oracle Internet Directory Server by Using Fusion Middleware Control

You can restart the Oracle Internet Directory Server using Fusion Middleware Control.

Restart the Oracle Internet Directory server as follows:

1. Go to the Oracle Internet Directory home page in Oracle Enterprise Manager Fusion Middleware Control.
2. From the Oracle Internet Directory menu, select **Control**, then **Restart**.
3. When the confirmation dialog appears, click **OK**.

If Fusion Middleware Control cannot restart the server, an error dialog appears.

8.3 Managing Oracle Internet Directory Components by Using WLST Commands

You can perform the following Oracle Internet Directory related tasks from the command line by using `WLST` Commands.

The following list of OID commands available for use can be obtained using `help('manageoid')` WLST command:

- Creating an Oracle Internet Directory Component by Using WLST Command — oid_createInstance
- Deleting an Oracle Internet Directory Component by Using WLST Command — oid_deleteInstance()
- Viewing Active Server Instance Information by Using WLST Command — oid_instanceStatus()
- Starting the Oracle Internet Directory Server by Using WLST Command — start()
- Stopping the Oracle Internet Directory Server by Using WLST Command — shutdown()
- Updating credential required by Enterprise Manager to manage OID - oid_setProperties()
- Fetching Enterprise Manager properties used to manage OID - oid_getProperties()
- Creating a Realm in Oracle Internet Directory -oid_createRealm()

- Listing all Oracle Internet Directory Instance Names-oid_listInstances()
- Updating orcladmin password - oid_setAdminPassword()

> **Note:**
>
> Arguments to `wlst` are case sensitive. Be sure to type them exactly as shown. For example, in the command `createInstance`, only the letter`I` is in upper case.
>
> For more information about options to an `WLST` command, type:
>
> ```
> wlst.sh
> ```
>
> ```
> help (command_name)
> ```

See Oracle Internet Directory Administration Tools in *Reference for Oracle Identity Management* for the syntax of the commands used in the examples.

8.3.1 Creating an Oracle Internet Directory Component by Using WLST Command — oid_createInstance

You can create an Oracle Internet Directory system component in an Oracle instance by using WLST Command: `oid_createInstance`.

> **Note:**
>
> Before executing the *oid_createInstance* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic*',password='*weblogic-password*',url='*t3://admin-server-host:admin-server-port')*

The syntax of `oid_createInstance` is:

```
oid_createInstance(instanceName='instance-name', machine='oidhost1', port = nnnn,
sslPort = nnnn, host = 'hostname')
```

Where:

- `instanceName` - This is the name of the managed instance being created.
- `machine` - This is the existing machine entry for the instance. You must specify `oidhost1` as the machine name.
- `orcladminPassword` - This is the password for super user `'cn=orcladmin'`.
- `port` - Optional. This is the port number of the non-SSL server. If this is not specified, a port will be assigned automatically.
- `sslPort` - Optional. This is the port number of the SSL virtual host. If this is not specified, a port will be assigned automatically.

- `host` - Optional. Name/IP address of the (logical) host, where OID server to be started/ stopped If not specified, hostname of the machine will be used.

The `oid_createInstance` command prompts for the WebLogic administrator's user name if you do not supply it. It also prompts for the passwords if you do not supply password file names on the command line. The `oid_createInstance` command also uses available ports if you do not specify `-port` or `-sslport`, as described in Oracle Internet Directory Ports.

8.3.2 Deleting an Oracle Internet Directory Component by Using WLST Command — oid_deleteInstance()

You can remove an Oracle Internet Directory component by using `oid_deleteInstance()` WLST Command. This also unregisters the component with the WebLogic server.

> **Note:**
>
> Before executing the *oid_createInstance* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic*',password='*weblogic-password*',url='*t3://admin-server-host:admin-server-port*')

The syntax of `oid_deleteInstance()` is:

```
oid_deleteInstance(instanceName = 'oid1')
```

Where,

`instanceName` is the name of the managed instance being deleted.

You are prompted for the WebLogic administrator's user name and password if you do not supply them.

8.3.3 Viewing Active Server Instance Information by Using WLST Command — oid_instanceStatus()

You can view the status of components and processes by using the WLST `oid_instanceStatus()` command.

To view the status, type:

```
oid_instanceStatus(instanceName = 'instance-name')
```

Where:

`instanceName` is the name of the OID instance.

```
Processes in Instance: asinst_2
---------------------------------+-------------------+---------+----------+------------
+----------+-----------+------
ias-component                    | process-type      |     pid | status   |        uid
|  memused |    uptime | ports-----------------------------------+--------------------
+---------+----------+------------+----------+-----------+------
```

```
oid2                            | oidldapd          |   24760 | Alive    |  988238800
|   102744 |   0:01:12 | N/A
oid2                            | oidldapd          |   24756 | Alive    |  988238799
|    55052 |   0:01:12 | N/A
oid2                            | oidmon            |   24745 | Alive    |  988238796
|    48168 |   0:01:14 | LDAPS:6789,LDAP:6788

oid1                            | oidldapd          |   21590 | Alive    |  988238048
|   103716 |  19:51:48 | N/A
oid1                            | oidldapd          |   21586 | Alive    |  988238047
|    54420 |  19:51:49 | N/A
oid1                            | oidmon            |   21577 | Alive    |  988238046
|    48168 |  19:51:49 | LDAPS:3133,LDAP:3060
```

8.3.4 Starting the Oracle Internet Directory Server by Using WLST Command — start()

You can start the Oracle Internet Directory Server using WLST `start()` command.

Note:

- Before executing the *start()* command, ensure that you connect to the weblogic server by using the *connect* command.
  The syntax for connecting to weblogic admin server is:

  connect(username='*weblogic',* password='*weblogic-password*', url='*t3://admin-server-host:admin-server-port')*
- Ensure that the Node Manager is up and running on the machine where you want to start Oracle Internet Directory instance.
- Alternatively, you can start Oracle Internet Directory instance using `startComponent.sh` command. Before executing `startComponent.sh` command, ensure that the Node Manager is up and running. You need not connect to WebLogic Server to execute `startComponent.sh` command. The syntax for `startComponent.sh` is:

  ```
  $DOMAIN_HOME/bin/startComponent.sh <instance-name>
  ```

  For an Oracle Autonomous Transaction Processing-Shared (ATP-S) database, before executing the `startComponent.sh` command:

  1. Set `TNS_ADMIN` property to `<$DOMAIN_HOME>/config/fmwconfig/components/OID/config/` using the following command:

     ```
     export TNS_ADMIN=<$DOMAIN_HOME>/config/fmwconfig/components/OID/config
     ```

     To update the contents of the directory pointed to by TNS_ADMIN, see Prerequisites for ATP-S.
  2. Start the Oracle Internet Directory instance:

     ```
     <$DOMAIN_HOME>/bin/startComponent.sh <instance-name>
     ```

The component name of the first Oracle Internet Directory component is `oid1`.

To start the first Oracle Internet Directory instance, type:

```
start(name='instance-name')
```

8.3.5 Stopping the Oracle Internet Directory Server by Using WLST Command — shutdown()

You can stop the Oracle Internet Directory server component using the WLST `shutdown()` command.

To stop the Oracle Internet Directory server component, type:

```
shutdown(name='instance-name')
```

8.3.6 Updating Credential Required by Enterprise Manager to manage OID - oid_setProperties()

Update the credentials for OID connection and ODSSM schema password for Enterprise Manager console to manage and monitor OID instances. This command is only relevant to collocated mode of OID installation where OID is manageable by Enterprise Manager.

> **Note:**
>
> - Before executing the *oid_setProperties()* command, ensure that you connect to the weblogic server by using the *connect* command.
>
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic'*, password='*weblogic-password'*, url='*t3://admin-server-host:admin-server-port')*
> - This command covers the functionality supported by `oidcred` tool that was used in previous release to update EMD and ODSSM passwords.

The syntax of `oid_setProperties()` is:

```
oid_setProperties(context='EM', host='host', port = nnnn, sslmode=nnn, sslwrl = 'file:/
wallet-location',
emdPassword = 'emd-login-password', odssmPassword = 'odssm-schema-password')
```

where,

- context - This is the context for which the properties are updated.

  Valid values:

  'EM' is for Enterprise Manager application context.
- `host` - Optional. Used in 'EM' context. OID host.
- `port` - Optional. Used in 'EM' context. OID port.
- `sslMode` - Optional. Used in 'EM' context. SSL mode.

  Valid values:

  – -1 : Non SSL mode.

  – 0 : SSL no auth mode (anonymous ciphers need to be enabled in OID)

  – 1 : SSL one way auth mode. sslwrl needs to be set.

  - 2 : SSL two way auth mode. sslwrl needs to be set.
- `sslwrl` - Optional. Wallet location.
- `emdPassword` - Optional. Used in 'EM' context.
  - Login password for EMD user (used by EM to connect to OID).
  - Password for EM user DN-`cn=emd admin,cn=oracle internet directory`
- `odssmPassword` - Optional. Used in 'EM' context. ODSSM schema password.

8.3.7 Fetching Enterprise Manager Properties Used to Manage OID - oid_getProperties()

Retrieves the Enterprise Manager properties used to manage OID. This command is only relevant to collocated mode of OID installation where OID is manageable by Enterprise Manager.

> **Note:**
>
> Before executing the *oid_getProperties()* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic',* password='*weblogic-password*', url='*t3://admin-server-host:admin-server-port')*

The syntax of `oid_getProperties()` is:

```
oid_getProperties(context='EM')
```

where,

`context` - This is the context for which the properties are retrieved.

Valid values: - 'EM' is for Enterprise Manage

This command returns the following values:

- Host = OID host
- Port = OID port
- sslMode = SSL mode
- sslwrl = wallet location

8.3.8 Creating a Realm in Oracle Internet Directory - oid_createRealm()

Creates a realm in Oracle Internet Directory.

> **Note:**
>
> Before executing the *oid_createRealm()* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic'*, password='*weblogic-password*', url='*t3://admin-server-host:admin-server-port')*

The syntax of `oid_createRealm()` is:

```
oid_createRealm(instanceName='instance-name', host='host-name', port = nnnn,
orcladminPassword = 'password', realmDN = 'namespace-name')
```

where,

- `instanceName` - This is the name of the managed OID instance
- `host` - Name/IP address of the OID host
- `port` - This is the port number of the OID
- `orcladminPassword` - This is the password for super user `'cn=orcladmin'`
- `realmDN` - This the new realm/namespace to be created

8.3.9 Listing all Oracle Internet Directory Instance Names - oid_listInstances()

Lists all Oracle Internet Directory instance names.

> **Note:**
>
> Before executing the *oid_listInstances()* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic'*, password='*weblogic-password*', url='*t3://admin-server-host:admin-server-port')*

The syntax of `oid_listInstances()` is: `oid_listInstances()`

8.3.10 Updating Orcladmin Password - oid_setAdminPassword()

This command updates the password for `orcladmin` super user.

> **Note:**
>
> Before executing the *oid_setAdminPassword()* command, ensure that you connect to the weblogic server by using the *connect* command.
> The syntax for connecting to weblogic admin server is:
>
> *connect*(username='*weblogic'*, password='*weblogic-password'*, url='*t3://admin-server-host:admin-server-port')*

The syntax of `oid_setAdminPassword` is:

```
oid_setAdminPassword(orcladminPassword = 'passwd', odsPassword = 'passwd')
```

where,

- `orcladminPassword` - New password for `cn=orcladmin`.
- `odsPassword`- DB password needed for verification.

8.4 Starting an Instance of the Replication Server by Using OIDCTL

You can configure an instance of Oracle Internet Directory Replication Server, using the `oidctl start` command with `server=oidrepld`. Best practice is to create a separate instance of Oracle Internet Directory to use for replication.

First create a new instance of Oracle Internet Directory as described in Creating Additional Oracle Internet Directory Instances. Then, ensure that the environment variable `DOMAIN_HOME` is set and type:

```
oidctl connect=connStr server=oidrepld inst=1 componentname=Component_Name \
   name=Instance_Name start
```

The `componentname` value must be the `component name` of the running `oidldapd` server. The `name` value must be the instance name of the running `oidldapd` server.

Do not start more than one instance of `oidrepld` on a host. Do not start `oidrepld` on more than one Oracle Internet Directory instance sharing the same Oracle Database.

> **Note:**
>
> The environment variables `DOMAIN_HOME`, `ORACLE_HOME`, and `COMPONENT_NAME` must be set before you run the `oidctl` command to start or stop the replication server.

See Also:

- Understanding Oracle Internet Directory Replication
- Setting Up Replication

9

Managing System Configuration Attributes

This chapter describes the configuration attributes that control the Oracle Internet Directory LDAP server and how to manage these attributes using Oracle Enterprise Manager Fusion Middleware Control, the WebLogic Scripting Tool (`wlst`), LDAP tools, and Oracle Directory Services Manager (ODSM).
For information about the attributes that control the Oracle Internet Directory replication server, see Managing Replication Configuration Attributes.

This chapter includes the following sections:

- Managing System Configuration Attributes
- Managing System Configuration Attributes by Using Fusion Middleware Control
- Managing System Configuration Attributes by Using WLST
- Managing System Configuration Attributes by Using LDAP Tools
- Managing System Configuration Attributes by Using ODSM Data Browser

> **See Also:**
>
> Application and System Configuration Attributes.

9.1 Managing System Configuration Attributes

Understand about managing various system configuration attributes.

This section contains the following topics:

- About Configuration Attributes
- About Operational Attributes
- Attributes of the Instance-Specific Configuration Entry
- Attributes of the DSA Configuration Entry
- Attributes of the DSE

9.1.1 About Configuration Attributes

Most Oracle Internet Directory configuration information is stored in the directory itself. The information is stored as attributes of specific configuration entries. You must have superuser privileges to set system configuration attributes.

Some configuration attributes are specific to an individual instance of the Oracle Internet Directory server. Instance-specific attributes are located in the instance-specific configuration entry, a specific subentry of the Oracle Internet Directory instance entry. Figure 8-1 shows the location of these entries in the DIT.

Some configuration attributes are shared by all Oracle Internet Directory server instances in a WebLogic Server domain that are connected to the same database. Shared attributes reside in the DSA Configuration entry. Replication-specific attributes reside in the Replica Subentry, Replication Configuration, and Replication Agreement Entry.

Some attributes reside in the DSE Root. Most of those are non-configurable.

See Understanding Process Control of Oracle Internet Directory Components

> **Note:**
>
> Oracle Internet Directory configuration attributes, either instance-specific or shared attributes, are not replicated. For example, computed attribute definitions from `OrclComputedAttribute` are stored in the DSA Configuration entry and are not replicated. If your deployment requires configuration attributes to be replicated, you must replicate them manually.

You can manage all the configuration attributes from the command-line. In addition, many of the configuration attributes have specific, task-oriented management interfaces in Oracle Enterprise Manager Fusion Middleware Control or Oracle Directory Services Manager. You can also use the Data Browser feature of Oracle Directory Services Manager to manage the entries directly.

> **See Also:**
>
> - Managing System Configuration Attributes by Using Fusion Middleware Control
> - Managing System Configuration Attributes by Using LDAP Tools
> - Managing System Configuration Attributes by Using ODSM Data Browser

9.1.2 About Operational Attributes

Do not confuse configuration attributes with operational attributes. Operational attributes have special meaning to the directory server and they are used for storing information needed for processing by the server itself or for holding other data maintained by the server that is not explicitly provided by clients. These are attributes that are maintained by the server and either reflect information the server manages about an entry or affect server operation.

Operational attributes are not returned by a search operation unless you specifically request them by name or with the "+" option in the search request. See Listing Operational Attributes by Using ldapsearch for more information.

Examples of operational attributes include the time stamp for an entry and the state values needed for enforcing password policies, described in Operational Attributes of User Entry. You cannot modify operational attributes.

From 11g Release 1 (11.1.1.9.0) onward, Oracle Internet Directory server returns `numsubordinate` operational attribute. It specifies the count of number of child entries under the given base DN.

> **Note:**
>
> By default the `numsubordinate` operational attribute is not returned when you specify the + option in the search request. You must explicitly set the `orcldseecompatible` flag to `1` in the `cn=dsaconfig,cn=configsets,cn=oracle internet directory` entry.

9.1.3 Attributes of the Instance-Specific Configuration Entry

During installation, Oracle Identity Management Installer creates an instance-specific configuration entry for the first Oracle Internet Directory instance.

It copies default values from a read-only entry under `cn=configset0`. (You can specify different values for the SSL port and non-SSL during the install.)

The DN of an instance-specific configuration entry has the form:

`cn=componentname,cn=osdldapd,cn=subconfigsubentry`

For example, if the component name for a server instance is `oid1`,then the DIT of the instance-specific configuration entry would be:

`cn=oid1,cn=osdldapd,cn=subconfigsubentry`

Table 9-1 lists the attributes of the instance-specific configuration entry. The **Update Mechanism** column contains the following abbreviations:

- `EM` – Oracle Enterprise Manager Fusion Middleware Control. See Managing System Configuration Attributes by Using Fusion Middleware Control.
- `WLST`–WebLogic Scripting tool. See Managing System Configuration Attributes by Using WLST.
- `LDAP`–LDAP command-line tools, such as `ldapmodify` and `ldapadd`. See Managing System Configuration Attributes by Using LDAP Tools.

Table 9-1 Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
`orclserverprocs`	Number of Server Processes. Restart the server after changing. See Understanding Process Control of Oracle Internet Directory Components.	EM, LDAP, WLST	1	Integer, up to 1024.
`orclreqattrcase`	Preserve the case of required attribute names specified in an `ldapsearch` request. See Getting Started With Oracle Internet Directory .	EM, LDAP	0	0: Do not preserve attribute case 1: Preserve attribute case

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclhostname	Hostname or IP address. See Managing IP Addresses in Oracle Internet Directory . See Managing Oracle Internet Directory Instances	LDAP	Set during install	Host or IP address
orclnonsslport	Non-SSL port See Configuring Server Properties. If you change the port number, restart the server. See Managing Oracle Internet Directory Instances.	EM, LDAP, WLST	3060	Port number
orclsslport	SSL port See Configuring Server Properties. If you change the port number, restart the server. See Managing Oracle Internet Directory Instances.	EM, LDAP, WLST	3131	Port number
orcltraceconndn	Distinguished name (DN) of a connection that causes Oracle Internet Directory server to log messages for operations performed by the specified connection DN, if orclDebugFlag is set to a value other than zero (0).	EM, LDAP, WLST	None	Multi-valued attribute that can specify one or more connection DNs.
orcltraceconnip	Connection IP address that causes Oracle Internet Directory server to log messages for operations performed by the specified connection IP address, if orclDebugFlag is set to a value other than zero (0).	EM, LDAP, WLST	None	Multi-valued attribute that can specify one or more connection IP addresses.
orcltxntimelimit	Maximum time allowed in a transaction (seconds). See Using LDAP Transactions in *Application Developer's Guide for Oracle Identity Management* and Configuring Server Properties.	EM, LDAP, WLST	0	Positive integer (seconds)
orcltxnmaxoperations	Maximum number of operations allowed in a transaction. See Using LDAP Transactions in *Application Developer's Guide for Oracle Identity Management* and Configuring Server Properties.	EM, LDAP, WLST	0	Positive integer

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclservermode	Server Mode See Performing Bulk Operations .	EM, LDAP, WLST	rw	R: read-only rw: read/write rm: read-modify
orclaudcustevents	A comma-separated list of events and category names to be audited. Custom events are only applicable when orclAudFilterPreset is Custom. See Managing Auditing .	EM, LDAP, WLST	Empty	Examples include: Authentication.SUCCESSESONLY, Authorization(Permission -eq 'CSFPermission')
orclaudfilterpreset	Replaces the audit levels used in 10g (10.1.4.0.1) and earlier releases. See Managing Auditing .	EM, LDAP, WLST	None	None, Low, Medium, All, and Custom.
orclaudsplusers	A comma separated list of users for whom auditing is always enabled, even if orclAudFilterPreset is None. See Managing Auditing .	EM, LDAP, WLST	Empty	Valid users. For example: cn=orcladmin
orclcachenotifyip	Associates a port number with an IP address in order to allow Oracle Internet Directory servers to communicate with each other in a cluster environment when cached data is changed.	LDAP	None	Port number and IP address See Configuring IP Addresses for Notifications in a Cluster.
orcldebugflag	Debug Flag See Managing Logging.	EM, LDAP, WLST	0	0 ~ 117440511 See Table 24-3.
orcldebugforceflush	Force flush debug messages See Managing Logging.	LDAP	0	0: Disable 1: Enable
orcldebugop	Operations Enabled for Debug See Managing Logging.	EM, LDAP, WLST	511	See Table 24-4.
orclmaxlogfiles	Maximum Number of Log Files to Keep in Rotation See Managing Logging.	EM, LDAP, WLST	100	Integer
orclmaxlogfilesize	Maximum Log File Size (MB) See Managing Logging.	EM, LDAP, WLST	1 MB	Size, in MB
orcleventlevel	Statistics collection event level See Monitoring Oracle Internet Directory.	EM, LDAP, WLST	0	See Table 25-5.
orcloptracklevel	Security event tracking level See Monitoring Oracle Internet Directory.	EM, LDAP, WLST	0	Table 25-3

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclstatsflag	Flag to turn on or off OID statistics data See Monitoring Oracle Internet Directory.	EM, LDAP, WLST	1	0: disable 1: enable
orclstatslevel	Enable user statistics collection See Monitoring Oracle Internet Directory.	EM, LDAP, WLST	0	0: disable 1: enable
orclstatsperiodicity	Frequency of flushing statistics to data bases See Monitoring Oracle Internet Directory.	EM, LDAP, WLST	30	60
orclsslauthentication	SSL Authentication Restart the server after changing. See Configuring Secure Sockets Layer (SSL).	EM, LDAP, WLST	1	1: No SSL authentication 32: One-way authentication 64: Two-way authentication
orclsslciphersuite	SSL Cipher Suite Restart the server after changing. See Configuring Secure Sockets Layer (SSL).	EM, LDAP, WLST	Empty	See the table in Supported Cipher Suites, left column.
orclsslenable	SSL Enable Restart the server after changing. Set orclsslenable to 1 or 2 if you use WLST or EM to configure the server. See Configuring Secure Sockets Layer (SSL).	EM, LDAP, WLST	2	0: Non-SSL only 1: SSL only, 2: Non-SSL & SSL mode
orclsslinteropmode	SSL Interoperability Mode Restart the server after changing. See Configuring Secure Sockets Layer (SSL).	LDAP	0	0: disabled 1: enabled
orclsslversion	SSL Version Restart the server after changing. See Configuring Secure Sockets Layer (SSL).	EM, LDAP, WLST	3	3
orclsslwalleturl	SSL Wallet URL Restart the server after changing. See Configuring Secure Sockets Layer (SSL).	EM, LDAP, WLST	File	SSL wallet file location.

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclanonymousbindsflag	Allow Anonymous binds See Managing Authentication ,	EM, LDAP, WLST	2	See Table 34-5.
orclsaslauthenticationmode	SASL Authentication Restart the server after changing Mode. See Managing Authentication .	EM, LDAP, WLST	1	auth, auth-int, auth-conf. Specify all three or a subset of these 3 as a comma separated string.
orclsaslcipherchoice	SASL Cipher Choice Restart the server after changing. See Managing Authentication .	EM, LDAP, WLST	Rc4-56,rc4-40,rc4,des, 3des	Any combination of Rc4-56, des, 3des, rc4, rc4-40
orclsaslmechanism	SASL Mechanism Restart the server after changing. See Managing Authentication .	EM, LDAP, WLST	DIGEST-MD5, EXTERNAL	DIGEST-MD5, EXTERNAL
orclmaskrealm	DIT Masking See Managing DIT Masking.	LDAP	No value	List of DIT subtrees.
orclmaskfilter	DIT Masking See Managing DIT Masking.	LDAP	No value	LDAP attribute filter.
orclmaskattribute	DIT Masking See Managing DIT Masking.	LDAP	No value	List of attributes, possibly preceded by **!**.
orcldispthreads	Maximum number of dispatcher threads per server process. See Oracle Internet Directory Performance Tuning in *Tuning Performance*. Restart server after changing.	EM, LDAP, WLST	1	Integer (Max 16)
orclldapconntimeout	LDAP Connection Timeout, in minutes See LDAP Server Attributes in *Tuning Performance*.	EM, LDAP, WLST	0	Integer Note: Users configured for statistics tracking do not time out as per this setting.
orclmaxcc	Maximum Number of DB Connections See Oracle Internet Directory Performance Tuning in *Tuning Performance*. Restart the server after changing.	EM, LDAP, WLST	2	Integer, maximum128

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclmaxconnincache	Maximum number of cached user group connections See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	EM, LDAP, WLST	100000	Integer
orclmaxldapconns	Maximum number of concurrent connections per server process See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	EM, LDAP, WLST	1024	Int (Max system max file descriptors per process)
orclmaxserverresptime	Maximum Time in seconds for Server process to respond back to Dispatcher process See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	EM, LDAP, WLST	0 seconds	Number of Seconds 0: Dispatcher does not detect the server hang.
orclnwrwtimeout	Maximum time in seconds for OID Server to wait for LDAP client respond to a Read/ Write operation. See Timeout for Write Operations in *Tuning Performance*.	EM, LDAP, WLST	30 seconds	Integer
orcloptrackmaxtotalsize	Maximum number of bytes of RAM that security events tracking can use for each type of operation. See Tuning Security Event Tracking in *Tuning Performance*.	LDAP	100000000 Bytes	Available RAM, in bytes
orcloptracknumelemcontainers Subtype: 1stlevel	Number of in-memory cache containers for storing information about users performing operations. See Tuning Security Event Tracking in *Tuning Performance*.	LDAP	256	Integer
orcloptracknumelemcontainers Subtype: 2ndlevel	Number of in-memory cache containers for storing information about users whose user password is compared and tracked when detailed compare operation statistics is programmed. See Tuning Security Event Tracking in *Tuning Performance*.	LDAP	256	Integer

Table 9-1 (Cont.) Attributes of the Instance-Specific Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
`orclpluginworkers`	Maximum number of plug-in worker threads per server process Restart the server after changing. See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	EM, LDAP, WLST	2	Int (Max 64)
`orclsizelimit`	Number of entries that can be returned in an `ldapsearch` result See Number of Entries to be Returned by a Search in *Tuning Performance*.	LDAP	10000	Integer
`orcltimelimit`	Maximum time that server can spend for a given `ldapsearch` operation	EM, LDAP, WLST	3600	Integer (seconds)
`orclsdumpflag`	Generate stack dump. See Troubleshooting Oracle Internet Directory.	LDAP	0	0: Generate stack trace file. 1: Do not generate stack. trace file, but generate a core file.
`orclskipspecialinfilter`	Evaluates whether Oracle Internet Directory should skip the processing of special characters specified in filter values during a search operation.	LDAP	0	0: Process the special characters specified in the filter value. 1: Do not process the special characters specified in the filter value.
`orclcryptoversion`	Allows you to specify the SSL/TLS version to be used.	LDAP	24	0: All Supported Protocols 2: For SSL v3.0 4: For TLS 1.0 8: For TLS 1.1 16: For TLS 1.2 24: For TLS 1.1 or TLS 1.2 **Note:** The attribute is additive in nature. This implies that it allows you to add more than one protocol by specifying the corresponding value. For more information, see Supported Protocol Versions.

9.1.4 Attributes of the DSA Configuration Entry

Understand about the attributes in the DSA configuration entry.

The DSA configuration entry has the DN:

```
cn=dsaconfig,cn=configsets,cn=oracle internet directory
```

Table 9-2 shows shared attributes in the DSA configuration entry. The **Update Mechanism** column contains the following abbreviations:

- `EM` – Oracle Enterprise Manager Fusion Middleware Control. See Managing System Configuration Attributes by Using Fusion Middleware Control.
- `LDAP`–LDAP command-line tools, such as `ldapmodify` and `ldapadd`. See Managing System Configuration Attributes by Using LDAP Tools.

> **Note:**
>
> DSA is an X.500 term for the directory server.

Table 9-2 Attributes in the DSA Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
`orclmaxpsearchconns`	Maximum number of connections allowed for an LDAP persistent search operation. See Persistent LDAP Search Operations.	EM, LDAP, WLST	0	Integer, up to 1024.
`orclblockdnip`	IP address that causes Oracle Internet Directory server to reject any new connections and close any existing connections from that IP address.	EM,LDAP	None	IP address
`orclcomputedattribute`	Mechanism to dynamically compute a configurable attribute and its value based on specific rules. See Managing Computed Attributes.	LDAP	None	Multi-valued attribute
`orclmaxlatencylog`	Time in microseconds after which any Oracle Internet Directory server operations that exceed this time are logged to the alert log.	EM,LDAP	10000000 microseconds. Minimum is 10 microseconds.	Microseconds
`orclmaxtcpidleconntime`	Frequency in minutes at which Oracle Internet Directory server calls `OCIPing()` to send keep alive messages to its Oracle Database. Setting this attribute to a value less than the timeout value of the firewall between Oracle Internet Directory server and the Oracle Database (typically 30 minutes) prevents the Database connection from being dropped.	LDAP	20 minutes	Integer 0: No `OCIPing()`

Table 9-2 (Cont.) Attributes in the DSA Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
	For zero downtime patching, `orclmaxtcpidleconntime`; ttl is set to 5 and admin is expected to wait for couple of cycles before turning off the database. After being done with number of cycles, it should be turned off with out of the box value 0. The value of this attribute is in minutes.	LDAP	0	Integer 0: Disabled
`orclmaxfiltsize`	Maximum Filter Size See Configuring Shared Properties.	EM, LDAP	24576	Integer
`orclrefreshdgrmems`	Refresh Dynamic Group Memberships. See Managing Dynamic and Static Groups in Oracle Internet Directory.	LDAP	0	1: Cause a refresh. Server will reset it to 0.
`orclautocatalog`	Index attributes on first search. See Index option in Oracle Internet Directory to Search Attributes .	EM, LDAP	1	0: Disabled 1: Enabled
`orclrienabled`	Referential Integrity. See Configuring Referential Integrity.	EM, LDAP	0	0: Disabled 1: Enabled
`orclstatsdn`	User DNs for statistics collection. See Monitoring Oracle Internet Directory.	EM, LDAP	Empty	DNs of entries
`orcldataprivacymode`	Sensitive attributes encrypted when returned See Configuring Data Privacy .	LDAP	0	0: Disabled 1: Enabled
`orclencryptedattributes`	Sensitive attributes stored in encrypted format. See Configuring Data Privacy .	LDAP	See Table 29-1.	Attributes
`orclhashedattributes`	Attributes stored in hashed format. See Configuring Data Privacy .	EM, LDAP	Empty	Attributes
`orclpkimatchingrule`	PKI Matching Rule for mapping user's PKI certificate DN to the user's entry DN. See Managing Authentication .	EM, LDAP	2	0: Exact match. 1: Certificate search. 2: Combination of 0 and 1. 3: Mapping rule only. 4: Try in order: 3, 2
`orclgeneratechangelog`	Whether to generate change logs for user operations. See Managing and Monitoring Replication and Oracle Internet Directory Performance Tuning in *Tuning Performance*.	LDAP	1	1: enable 0: disable

Table 9-2 (Cont.) Attributes in the DSA Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orcljvmoptions	Options passed to the JVM when a server plug-in is invoked. See Developing Plug-ins for the Oracle Internet Directory Server.	EM, LDAP	-Xmx64M	Valid JVM options
orclinmemfiltprocess	Search Filters to be processed in memory See Optimizing Performance of Complex Search Filters in *Tuning Performance*.	EM, LDAP	See list in *Tuning Performance*	Valid search filters
orclmatchdnenabled	Whether to provide detailed MatchDN information when base DN of a search is not present. See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	EM, LDAP	1	0: Do not match, but validates if baseDN exists in the database 1: Match 2: Perform no DB check for existence of base DN
orclskewedattribute	Skewed attributes. Server restart recommended after changing. See Optimizing Searches for Skewed Attributes in *Tuning Performance*.	EM, LDAP	objectclass	List of attributes
orclskiprefinsql	Skip referral for search. Server restart recommended after changing. See LDAP Server Attributes in *Tuning Performance*.	EM, LDAP	0	0: Disabled 1: Enabled
orcltlimitmode	Specify search time limit mode to be either accurate or approximate. See Oracle Internet Directory Performance Tuning in *Tuning Performance*.	LDAP	0	0: Accurate 1: Approximate
orclcachemaxsize	Size in bytes of the Result Set cache or Metadata cache, as indicated by the subtype (rs or md). Requires a server restart to take effect.	LDAP	Result Set cache: 64 MB (64 MB is also the minimum cache size) Metadata cache: 128 MB (128 MB is also the minimum cache size).	Subtype: rs (Result Set cache) or md (Metadata cache) Size: M (megabytes) or G (gigabytes).

Table 9-2 (Cont.) Attributes in the DSA Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclecacheenabled	Enable or disable the Entry Cache or Result Set Cache.. See Server Entry Cache in *Tuning Performance*.	EM, LDAP, WLST	2	0: Disable both caches 1: Enable Entry Cache only 2: Enable both caches 4: Pre-load cache data during server start up time or when cache is destroyed. Oracle Internet Directory servers rebuild the cache when orclecacheenabled is set to 4. **Note:** Entry cache pre-load is based on orclrscacheattr settings.
orclecachemaxentries	Maximum Entries in Entry Cache. See Server Entry Cache in *Tuning Performance*.	EM, LDAP, WLST	100000	Integer
orclecachemaxsize	Entry Cache Size in bytes. See Server Entry Cache in *Tuning Performance*.	EM, LDAP, WLST	1024 Megabytes or 1 Gigabyte	Size: M (megabytes) or G (gigabytes). For example: 1024M
orclrscacheattr	Result Set Cache Attributes See Result Set Cache in *Tuning Performance*.	EM, LDAP, WLST	cn uid mail orclguid	Multi-valued attribute that specifies the Result Set Cache attributes. Typically these attributes are not modified for the life of the entry. If an attribute has referential integrity enabled, that attribute should not be used.
orclenablegroupcache	Enable/Disable Group cache See Enabling the Group Cache in *Tuning Performance*.	LDAP	1	1: Enable, 0: Disable
orcldseecompatible	If orcldseecompatible is set to 1, then the attribute numsubordinates is returned if the search request has "+" required attribute.		None	

Table 9-2 (Cont.) Attributes in the DSA Configuration Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclqosconfig	Mechanism to dynamically configure throttling polices. See Managing Quality of Service Configuration.	LDAP	None	Multi-valued attribute

9.1.5 Attributes of the DSE

The DSA-specific entry (DSE) is the root of the DIT. This is where Oracle Internet Directory publishes information about itself, such as naming contexts, supported controls, and matching rules. Most attributes of the DSE should not be modified directly.

> **Note:**
>
> The orclcompatibleversion DSE attribute contains the Oracle Internet Directory version. This attribute is multi-valued. The values can be:
>
> ```
> orclcompatibleversion=OID 11.1.1.6.0
> orclcompatibleversion=OID 11.1.1.7.0
> orclcompatibleversion=OID 11.1.1.9.0
> orclcompatibleversion=OID 12.2.1.3.0
> orclcompatibleversion=OID 12.2.1.4.0
> orclcompatibleversion=OID 12.2.1.5.0
> orclcompatibleversion=OID 12.2.1.6.0
> orclcompatibleversion=OID 14.1.2.0.0
> orclcompatibleversion=OID 14.1.2.1.0
> ```
>
> Do not modify orclcompatibleversion. It must be present for Oracle Internet Directory to work with its respective schema.

Some DSE attributes that you might need to modify are listed in Table 9-3.

Table 9-3 Attributes of the DSE

Attribute	Description	Update Mechanism	Default	Possible Values
namingcontexts	Naming contexts. See Managing Naming Contexts in Oracle Internet Directory.	LDAP	c=us dc=com	Any valid naming context.
ref	Referral specification. See Managing Knowledge References and Referrals.	LDAP		

Table 9-3 (Cont.) Attributes of the DSE

Attribute	Description	Update Mechanism	Default	Possible Values
`orclaci`	Access control at the root DSE level. See Managing Directory Access Control.	LDAP		
`orclcryptoscheme`	Hashing algorithm for protecting passwords. See Managing Password Verifiers .	LDAP	SSHA	`SHA, SSHA, SHA256, SHA384, SHA512, SSHA256, SSHA384, SSHA512, SMD5, UNIX Crypt`
`subentry`	Contains DN of password policy governing the DSE root. See Managing Password Policies .	LDAP	cn=default,cn=pwdPolicies,cn=Common, cn=Products,cn=OracleContext	
`orclsimplemodchglogattributes`	List of multivalued attributes for which change logs contain only changes, not lists of all values. See Change Logs in Directory Replication.	LDAP	`member, uniqueMember`	Multivalued attributes

9.2 Managing System Configuration Attributes by Using Fusion Middleware Control

You can view and set most of the configuration attributes for an Oracle directory server by using Oracle Enterprise Manager Fusion Middleware Control.

This section contains the following topics:

- Configuring Server Properties
- Configuring Shared Properties
- SSL and Audit Parameters Configuration

9.2.1 Configuring Server Properties

You can configure attributes using the Oracle Internet Directory Server Properties pages of Fusion Middleware Control. The various options in the Server Properties pages, such as, General and Performance are listed in the following sections.

This section includes the following topics:

- Configuring Server Properties
- General Options in Configuring Server Properties
- Performance Options in Configuring Server Properties
- SASL Tab of Server Properties
- Statistics Tab of Server Properties

- Logging Tab of Server Properties

9.2.1.1 Configuring Server Properties

You can configure most of the attributes in the instance-specific configuration entry by using the Oracle Internet Directory **Server Properties** pages of Fusion Middleware Control as follows:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu.
2. Select **General**, **Performance**, **SASL**, **Statistics**, or **Logging**, depending on which parameters you want to configure.
3. After changing the configuration, choose **Apply**.

9.2.1.2 General Options in Configuring Server Properties

The correspondence between server properties and configuration attributes on the General tab of the Server Properties page is shown in Table 9-4.

Table 9-4 Configuration Attributes on Server Properties Page, General Tab.

Field or Heading	Configuration Attribute
Server Mode	`orclservermode`
Maximum number of entries to be returned by a search	`orclsizelimit`
Maximum time allowed for a search to complete (sec)	`orcltimelimit`
Preserve Case of Required Attribute Name specified in Search Request	`orclreqattrcase`
Anonymous Bind	`orclanonymousbindsflag`
Maximum time allowed in a Transaction (sec)	`orcltxntimelimit`
Maximum Number of Operations allowed in a Transaction	`orcltxnmaxoperations`
Non-SSL Port	`orclnonsslport`
SSL Port	`orclsslport`

9.2.1.3 Performance Options in Configuring Server Properties

The correspondence between server properties and configuration attributes on the Performance tab of the Server Properties page is shown in Table 9-5

Table 9-5 Configuration Attributes on Server Properties Page, Performance Tab

Field or Heading	Configuration Attribute
Number of OID LDAP Server Processes	`orclserverprocs`
Number of DB Connections per Server Process	`orclmaxcc`
Number of users in privilege Group membership Cache	`orclmaxconnincache`
LDAP Idle Connection Timeout (minutes)	`orclldapconntimeout`
OID server Network Read/Write Retry Timeout (sec)	`orclnwrwtimeout`
Maximum Number of LDAP connections per Server Process	`orclmaxldapconns`

Table 9-5 (Cont.) Configuration Attributes on Server Properties Page, Performance Tab

Field or Heading	Configuration Attribute
Maximum Time in seconds for Server process to respond back to Dispatcher process	`orclMaxServerRespTime`
Number of Dispatcher Threads per Server Process	`orcldispthreads`
Number of Plug-in Threads per Server Process	`orclpluginworkers`
Enable Change Log Generation	`orclgeneratechangelog`

Restart the server after changing `orclserverprocs`, `orclmaxcc`, `orcldispthreads`, or `orclpluginworkers`.

9.2.1.4 SASL Tab of Server Properties

The correspondence between server properties and configuration attributes on the SASL tab of the Server Properties page is shown in Table 34-2.

9.2.1.5 Statistics Tab of Server Properties

The correspondence between server properties and configuration attributes on the Statistics tab of the Server Properties page is shown in Table 25-2.

9.2.1.6 Logging Tab of Server Properties

The correspondence between server properties and configuration attributes on the Logging tab of the Server Properties page is shown in Table 24-2.

9.2.2 Configuring Shared Properties

You can configure some of the shared system configuration attributes in the DSA configuration entry by using the Oracle Internet Directory **Shared Properties** page of Fusion Middleware Control.

This section contains the following topics:

- Configuring Shared Properties
- Configuration Attributes in General Properties
- Change Superuser Password
- Replication

9.2.2.1 Configuring Shared Properties

To configure some of the shared system configuration attributes in the DSA configuration entry, select **Administration**, then **Shared Properties**, then select **General**, **Change Superuser Password**, or **Replication** from the **Oracle Internet Directory** menu. After changing the configuration, choose **Apply**.

9.2.2.2 Configuration Attributes in General Properties

Table 9-6 lists the configuration attributes available in the General Tab on the Shared Properties Tab.

Table 9-6 Configuration Attributes on Shared Properties Page, General Tab

Field or Heading	Configuration Attribute
User DN	`orclstatsdn`
Skip referral for search	`orclskiprefinsql`
Skewed attributes	`orclskewedattribute`
Search Filters to be processed in memory	`orclinmemfiltprocess`
Hashed attributes	`orclhashedattributes`
Match DN	`orclMatchDnEnabled`
PKI Matching Rule	`orclPKIMatchingRule`
Referential Integrity	`orclrienabled`
Maximum Filter Size	`orclmaxfiltsize`
Enable Entry Cache	`orclecacheenabled`
Maximum Entries in Entry Cache	`orclecachemaxentries`
Maximum Entry Cache Size (MB)	`orclecachemaxsize`
Number of users in privilege group membership cache NOT on EM page	`orclmaxconnincache`
Result Set Cache Attributes	`orclrscacheattr`
Java Plug-in VM Options	`orcljvmoptions`

A server restart is recommended after changing `orclskiprefinsql` or `orclskewedattribute`.

9.2.2.3 Change Superuser Password

See Changing the Superuser Password by Using Fusion Middleware Control.

9.2.2.4 Replication

Replication-related attributes are described in Managing Replication Configuration Attributes.

9.2.3 SSL and Audit Parameters Configuration

You can configure SSL parameters by using the Oracle Internet Directory SSL Configuration Page.

See Overview of Configuring SSL by Using Fusion Middleware Control. You must restart the server for SSL configuration changes to take effect.

You can configure Audit attributes by using the Oracle Internet Directory Audit Policy Settings page. See Managing Auditing Using Fusion Middleware Control.

9.3 Managing System Configuration Attributes by Using WLST

You can manage system configuration attributes using WLST.

Table 9-7 lists the Related MBeans.

This section includes the following topics:

- Managing System Configuration Attributes Using WLST
- Related MBeans Of Oracle Internet Directory

9.3.1 Managing System Configuration Attributes Using WLST

You can use the WebLogic Scripting Tool (`wlst`) in the Oracle Common home to manage the attributes of the Oracle Internet Directory instance-specific configuration entry that have Oracle Enterprise Manager Fusion Middleware Control interfaces.

A managed bean (MBean) is a Java object that represents a JMX manageable resource in a distributed environment, such as an application, a service, a component or a device. The WebLogic server uses custom MBeans as its interface to system components, such as Oracle Internet Directory.

> **Note:**
>
> WLST manages Oracle Internet Directory through its SSL port. The Oracle Internet Directory SSL port must be configured for no authentication or server authentication. If the Oracle Internet Directory SSL port is configured for mutual authentication, you will not be able to change Oracle Internet Directory attributes by using WLST. See About SSL Authentication Modes.

> **See Also:**
>
> - Oracle Fusion Middleware Components in *Administering Oracle Fusion Middleware*
> - Using the WebLogic Scripting Tool in *Understanding the WebLogic Scripting Tool*
> - Managing Auditing Using WLST

To use WLST, follow the steps below:

1. Invoke WLST

   ```
   $ORACLE_HOME/oracle_common/common/bin/wlst.sh
   ```

2. Connect to the WebLogic server

   ```
   connect('username', 'password', 't3://localhost:7001')
   ```

3. To navigate to the custom mbean tree, type:

   ```
   custom()
   ```

at the wlst prompt.

4. To get a one-level list of the MBean in the custom MBean tree, type:

```
ls()
```

In the `ls()` output, you see two domains that contain MBeans that are related to Oracle Internet Directory configuration. The domains are `oracle.as.management.mbeans.register` and `oracle.as.oid`.

5. To get to a domain, use the `cd()` command. For example:

```
cd('oracle.as.management.mbeans.register')
```

or

```
cd('oracle.as.oid')
```

If you type `ls()`, you see a list of MBeans in that domain. There are three MBeans related to Oracle Internet Directory configuration under `oracle.as.management.mbeans.register` and two under `oracle.as.oid`. Table 9-7 lists them.

INSTANCE and *COMPONENT_NAME* refer to the Oracle instance where your Oracle Internet Directory component is located and the name of the component, respectively.

> **Note:**
>
> The Audit MBean is shown here for completeness, but you use different commands for managing auditing by using `wlst`. See Managing Auditing Using WLST.

6. To get to a specific MBean, type:

```
cd('MBEAN_NAME')
```

For example, if you are in the domain `oracle.as.management.mbeans.register`, and you want to manage the Root Proxy MBean for Oracle Internet Directory component `oid1` in Oracle instance `instance1`, type:

```
cd('oracle.as.management.mbeans.register:type=OID,name=oid1,instance=instance1')
```

7. Once you have navigated to the desired MBean, you can get the current value for an attribute by typing:

```
get('ATTRIBUTE_NAME')
```

For example, to get the value for `orclserverprocs`, type:

```
get('orclserverprocs')
```

8. Before you make any changes to attributes, you must ensure that the MBean has the current server configuration. To do that, load the configuration from Oracle Internet Directory server to the mbean. Type:

```
invoke('load',jarray.array([],java.lang.Object),jarray.array([],java.lang.String))
```

9. Then you can use the set command to set a specific attribute. Type:

```
set('ATTRIBUTE_NAME', ATTRIBUTE_VALUE)
```

For example, to set `orclserverprocs` = `12`, type:

```
set('orclserverprocs', 12)
```

10. After making changes, you must save the MBean configuration to the Oracle Internet Directory server. Type:

```
invoke('save',jarray.array([],java.lang.Object),jarray.array([],java.lang.String))
```

9.3.2 Related MBeans Of Oracle Internet Directory

There are three MBeans related to Oracle Internet Directory configuration under `oracle.as.management.mbeans.register` and two under `oracle.as.oid`.

Table 9-7 lists all the MBeans.

Table 9-7 Oracle Internet Directory-Related MBeans

MBean Name	MBean Domain	MBean Format in ls() Output
Root Proxy MBean	`oracle.as.management.mbeans.register`	`oracle.as.management.mbeans.register:type=component,name=`*`COMPONENT_NAME`*`,instance=`*`INSTANCE`*
Non-SSL Port MBean	`oracle.as.management.mbeans.register`	`oracle.as.management.mbeans.register:type=component.nonsslport,name=nonsslport1,instance=`*`INSTANCE`*`,component=`*`COMPONENT_NAME`*
Audit MBean	`oracle.as.management.mbeans.register`	`oracle.as.management.mbeans.register:type=component.auditconfig,name=auditconfig1,instance=`*`INSTANCE`*`,component=`*`COMPONENT_NAME`*
SSL Port MBean	`oracle.as.oid`	`oracle.as.oid:type=component.sslconfig,name=sslport1,instance=`*`INSTANCE`*`,component=`*`COMPONENT_NAME`*
Key Store MBean	`oracle.as.oid`	`oracle.as.oid:type=component.keystore,name=keystore,instance=`*`INSTANCE`*`,component=`*`COMPONENT_NAME`*

9.4 Managing System Configuration Attributes by Using LDAP Tools

From the command line, you can modify most system configuration attributes by using `ldapmodify` and list most system configuration by using `ldapsearch`.

This section describes:

- Setting System Configuration Attributes by Using ldapmodify
- Listing Configuration Attributes with ldapsearch

9.4.1 Setting System Configuration Attributes by Using ldapmodify

You can modify system configuration attributes using ldapmodify.

You can modify most attributes in Table 9-1, Table 9-2, and Table 9-3 by using the command-line:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

The contents of the LDIF file depends on the DN and the operation being performed.

The LDIF file for changing the value of the `orclgeneratechangelog` attribute in the instance-specific entry to `1` would be:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclgeneratechangelog
orclgeneratechangelog: 1
```

The LDIF file for adding the `orclinmemfiltprocess` attribute to the DSA configuration entry would be:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype: modify
add: orclinmemfiltprocess
orclinmemfiltprocess: (objectclass=inetorgperson)(orclisenabled=TRUE)
```

Note:

- Since 11g Release 1 (11.1.1.0.0), consecutive settings of `orcldebugflag` and of `orcloptracklevel` are additive.
- Restart the server after changing `orclskiprefinsql`, `orclskewedattribute`, `orclserverprocs`, `orcldispthreads`, `orclmaxcc`, `orclpluginworkers`, or any attribute with a name that begins with "`orclssl`" or "`orclsasl`."
- After changing `orclnonsslport` or `orclsslport`, restart the server.

See Also:

- The Oracle Internet Directory Performance Tuning chapter in *Tuning Performance* for more examples of LDIF files.
- The command-line tool reference, `ldapmodify` in *Reference for Oracle Identity Management* for a more detailed discussion of `ldapmodify`, and a list of its options.

  .
- The "Oracle Identity Management " LDAP Attribute Reference in *Reference for Oracle Identity Management* for descriptions of the modifiable system configuration attributes.

9.4.2 Listing Configuration Attributes with ldapsearch

You can use `ldapsearch` to list most attributes.

For example:

- Instance-Specific Configuration Entry

If the component name for a server instance is `oid1`,then you can list the attributes in the instance-specific configuration entry with a command line such as:

```
ldapsearch -p 3060 -h myhost.example.com -D cn=orcladmin -q \
   -b "cn=oid1,cn=osdldapd,cn=subconfigsubentry" -s base "objectclass=*"
```

- DSA Configuration Entry

  You can list the attributes with the command line:

  ```
  ldapsearch -p 3060 -h myhost.example.com -D cn=orcladmin -q \
     -b "cn=dsaconfig,cn=configsets,cn=oracle internet directory" \
     -s base "objectclass=*"
  ```

- DSE

  You can list the attributes with the command line:

  ```
  ldapsearch -p 3060 -h myhost.example.com  -D cn=orcladmin -q \
      -b "" -s base "objectclass=*"
  ```

9.5 Managing System Configuration Attributes by Using ODSM Data Browser

Oracle Enterprise Manager Fusion Middleware Control is the recommended graphical user interface for managing system configuration attributes. You can also use ODSM to manage system configuration attributes, which can be useful if Fusion Middleware Control is not available or if you must modify an attribute that has no Fusion Middleware Control interface.

See Managing Entries by Using Oracle Directory Services Manager for detailed instructions for changing the attributes of a directory entry. The following sections explain how to get to the entries that contain system configuration attributes in ODSM.

This section includes the following topics:

- Navigating to the Instance-Specific Configuration Entry
- Navigating to the DSA Configuration Entry
- Navigating to the DSE Root

9.5.1 Navigating to the Instance-Specific Configuration Entry

You can navigate to the Instance-specific configuration entry from the ODSM Data Browser tab.

On the Data Browser tab, in the navigation tree, expand `subconfigsubentry`, then `osdldapd`. Then select the name of the Oracle Internet Directory component you want to manage.

9.5.2 Navigating to the DSA Configuration Entry

You can navigate to the DSA configuration entry from the ODSM Data Browser tab.

On the Data Browser tab, in the navigation tree, expand `oracle internet directory`, then `configsets`, then select the entry `dsaconfig`.

9.5.3 Navigating to the DSE Root

You can navigate to the DSE root from the ODSM Data Browser tab.

On the Data Browser tab, click `Root` in the navigation tree to select the DSE.

10

Managing IP Addresses in Oracle Internet Directory

Understand about the Oracle Internet Directory IP addresses and how to configure these IP addresses for IPV6, cold failover cluster, virtual IP, or notifications in a cluster.

This section contains the following topics:

- Introduction to Managing IP Addresses
- Configuring an IP Address for IPV6, Cold Failover Cluster, or Virtual IP
- Configuring IP Addresses for Notifications in a Cluster

10.1 Introduction to Managing IP Addresses

When you install Oracle Internet Directory on a dual stack (IPV4/IPV6) host, Oracle Internet Directory listens on both addresses. You cannot install Oracle Internet Directory on a host with only an IPV6 address because the Oracle Database requires an IPV4 address to connect to.

If you install Oracle Internet Directory on an IPV4 host and then change the host's address to IPV6, you must configure Oracle Internet Directory's IP address separately to the IPV6 address by changing the `orclhostname` attribute in the instance-specific configuration entry.

If you must have Oracle Internet Directory listen on a specific address for some other reason, you also do that by changing the `orclhostname` attribute in the instance-specific configuration entry.

10.2 Configuring an IP Address for IPV6, Cold Failover Cluster, or Virtual IP

You can configure an IP address using `ldapmodify` command.

To configure Oracle Internet Directory to listen on a specific IP address:

1. Create an LDIF file similar to this:

   ```
   dn: cn=COMPONENT_NAME, cn=osdldapd, cn=subconfigsubentry
   changetype: modify
   replace: orclhostname
   orclhostname: IP_address
   ```

2. Execute the following `ldapmodify` command:

   ```
   ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
   ```

3. Restart Oracle Internet Directory by using `wlst` command, as follows:

   ```
   $DOMAIN_HOME/bin/startComponent.sh <instance-name>
   ```

You can also use ODSM to change the `orclhostname` attribute in the instance-specific configuration entry. See Managing System Configuration Attributes by Using ODSM Data Browser.

10.3 Configuring IP Addresses for Notifications in a Cluster

Understand how to configure a dedicated IP addresses and Oracle Internet Directory instance for notifications in a cluster environment.

This section contains the following topics:

- Configuring a Dedicated IP Address and Oracle Internet Directory Instance for Notifications
- Configuring an Oracle Internet Directory Instance with an IP Address
- Configuring an Oracle Internet Directory Instance with a Port Number

10.3.1 Configuring a Dedicated IP Address and Oracle Internet Directory Instance for Notifications

In a cluster environment, Oracle Internet Directory servers need to communicate with each other when cached data is changed. These servers communicate using the LDAP protocol.

The cluster environment now supports SSL communication. The following points should be kept in mind:

- If both non-SSL and SSL ports are enabled, communication will occur over non-SSL port.
- If only SSL ports are enabled, communication will occur over SSL, provided the certificates for remote OID instances are correctly imported into the OID wallet, as defined by the **orclsslwalleturl** attribute in the instance configuration entry. If the certificates are not properly configured, cluster communication will fail.

For better performance, use a dedicated IP address and Oracle Internet Directory instance for notifications. Create a new component and then add the instance configuration, as follows:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f ldiffile
```

In this example, ldiffile contains:

```
dn: cn=oid-instance,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclcachenotifyip;port-number-to-use
orclcachenotifyip;port-number-to-use: IP-address-to-use
```

where:

- *oid-instance* is the dedicated Oracle Internet Directory component, such as oid1, oid2, or oid3.
- *port-number-to-use* is the port number you want to use for notifications.
- *IP-address-to-use* is the IP address you want to use for notifications.

10.3.2 Configuring an Oracle Internet Directory Instance with an IP Address

You can configure an Oracle Internet Directory Instance with an IP address using `ldapmodify` command.

For example, on node 1 if you have two IP addresses 10.10.10.1 and 10.10.10.2 and you want to use 10.10.10.2 for notifications, perform the following configuration, where oid1 is the component name on node 1:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f ldiffile
```

In this example, ldiffile contains:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclcachenotifyip
orclcachenotifyip: 10.10.10.2
```

Similarly on node 2, if you have two IP addresses 10.10.10.3 and 10.10.10.4 and if you want to use 10.10.10.4 for notifications, perform the following configuration, where oid2 is the component name on node 2:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f ldiffile
```

In this example, ldiffile contains:

```
dn: cn=oid2,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclcachenotifyip
orclcachenotifyip: 10.10.10.4
```

> **Note:**
>
> When `orclcachenotifyip` is configured for an Oracle Internet Directory instance, the IP address must be local to the node where that instance is running.
>
> For example, if Oracle Internet Directory on node 1 is `cn=oid1,cn=osdldapd,cn=subconfigsubentry` and Oracle Internet Directory on node 2 is cn`=oid2,cn=osdldapd,cn=subconfigsubentry`, then Oracle Internet Directory on node 1 will check configuration information of node 2 (which is `cn=oid2,cn=osdldapd,cn=subconfigsubentry`).
>
> When Oracle Internet Directory server on node 1 is started, it will use this information to connect to Oracle Internet Directory on node 2 (10.10.10.4).

10.3.3 Configuring an Oracle Internet Directory Instance with a Port Number

Understand how to configure an Oracle Internet Directory Instance with a Port Number. If you do not want production traffic to be affected with notification LDAP traffic, then create a new OID instance on each node.

For example, on node 1, create oid3 with port number 5678, as follows:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f ldiffile
```

In this example, ldiffile contains:

```
dn: cn=oid3,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclcachenotifyip;5678
orclcachenotifyip;5678: 10.10.10.2
```

Repeat the same procedure on node 2 to create the oid4 instance with port number 5678, as follows:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f ldiffile
```

In this example, ldiffile contains:

```
dn: cn=oid4,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclcachenotifyip;5678
orclcachenotifyip;5678: 10.10.10.4
```

11

Managing Naming Contexts in Oracle Internet Directory

The following topics describe Oracle Internet Directory naming contexts, including how to search for naming contexts and how to publish naming contexts to enable users to search for specific naming contexts:

- Publishing Naming Contexts
- Searching for Published Naming Contexts
- Modifying a Naming Context

See "Directory Naming Contexts" for a description of naming contexts.

11.1 Publishing Naming Contexts

To publish a naming context, you specify the topmost entry of each naming context as a value of the `namingContexts` attribute in the root DSE. For example, suppose you have a DIT with three major naming contexts, the topmost entries of which are `c=uk`, `c=us`, and `c=de`.

If these entries are specified as values in the `namingContexts` attribute, then a user, by specifying the appropriate filter, can find information about them by searching the root DSE. The user can then focus the search—for example, by concentrating on the `c=de` naming context in particular.

11.2 Searching for Published Naming Contexts

To search for published naming contexts, perform a base search on the root DSE with `objectClass =*` specified as a search filter. The retrieved information includes those entries specified in the `namingContexts` attribute.

For example:

```
ldapsearch -p 3060 -q -D cn=orcladmin -b "" -s base -L "objectclass=*" \
namingcontexts
```

> **Note:**
>
> This command will not return anything unless naming contexts have been published.

Before you publish a naming context, be sure that:

- You are a directory administrator with the necessary access to the root DSE.
- The topmost entry of that naming context exists in the directory.

11.3 Modifying a Naming Context

You use ldapmodify to publish a naming context. The `namingContexts` attribute is multi-valued, so you can specify multiple naming contexts.

To modify `namingContexts` by using the command-line:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

The following sample LDIF file specifies the entry `c=uk` as a naming context.

```
dn:
changetype: modify
add: namingcontexts
namingcontexts: c=uk
```

12

Managing Accounts and Passwords in Oracle Internet Directory

You can manage Oracle Internet Directory accounts and passwords using command-line tools, Self-Service Console, Oracle Directory Services Manager, and Oracle Enterprise Manager Fusion Middleware Control and also you can manage passwords for a superuser account, the EMD administrator, and the Oracle Internet Directory database.
The following topics describe managing accounts and passwords in Oracle Internet Directory:

- Introduction to Managing Accounts and Passwords
- Managing Accounts and Passwords by Using Command-Line Tools
- Managing Accounts and Passwords by Using the Self-Service Console
- Unlocking Locked Accounts by Using Oracle Directory Services Manager
- Changing the Superuser Password by Using Fusion Middleware Control
- Creating Another Account With Superuser Privileges
- Managing the Superuser Password by Using ldapmodify
- Changing the Oracle Internet Directory Database Password
- Resetting the Superuser Password
- Changing the Password for the EMD Administrator Account
- Changing the Password for the ODSSM Administrator Account
- Updating the New ODSSM Password for Data Source

12.1 Introduction to Managing Accounts and Passwords

Using command-line tools or the Self-Service console, you can perform administrative tasks related to account and passwords.

See Also:

- Managing Directory Entries in Oracle Internet Directory
- Managing Password Policies

Note:

To manage users using Self-Service console in Oracle Identity Manager, See Managing Users in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager*

Using command-line tools or the Self-Service console, you can temporarily disable a user's account, then enable it again. If you are a member of the Security Administrators Group, then you can unlock an account without resetting the user password. This saves you from having to explicitly tell the user the new password. The user can simply log in using the old password.

Using command-line tools, you can force users to change their passwords when they log in for the first time.

If you forget your password or become locked out of your account, then you can reset your password. You do this by using the Self-Service Console. This involves identifying yourself to the server by providing values for a set of password validation attributes. This takes the form of answering a password hint question to which you had earlier specified an answer.

The Superuser is a special directory administrator with full access to directory information. The default user name of the superuser is `orcladmin`. The password is set by the administrator during installation.

> **Note:**
>
> Oracle recommends that you change the password immediately after installation.

You can use either Oracle Enterprise Manager or `ldapmodify` to administer the Superuserpassword.

> **See Also:**
>
> Managing Directory Access Control for information on how to set access rights

Another privileged account is the administrator, `"cn=emd admin,cn=oracle internet directory"`. This account is used for starting and stopping Oracle Internet Directory server manageability information collection. It is also used by Oracle Enterprise Manager Fusion Middleware Control to make configuration changes to Oracle Internet Directory. These changes are made over a secure connection.

The only way you can change this account's password is to use the procedure documented in Changing the Password for the EMD Administrator Account. There is no support in the `oidpasswd` tool for changing this password.

12.2 Managing Accounts and Passwords by Using Command-Line Tools

You can perform admin operations on user accounts by using command-line tools.

This section contains these topics:

- Enabling and Disabling Accounts by Using Command-Line Tools
- Unlocking Accounts by Using Command-Line Tools
- Forcing a Password Change by Using Command-Line Tools

12.2.1 Enabling and Disabling Accounts by Using Command-Line Tools

You can temporarily disable a user's account, then enable it again, by using command-line tools.

To permanently disable the account, set the `orclisenabled` attribute to `DISABLED`. Setting this attribute to any other value enables the account.

To enable the account after you have disabled it, delete this attribute from the entry.

To enable the account for a specific period, set the `orclActiveStartDate` and `orclActiveEndDate` attributes in the user entry to the proper value in UTC (Coordinated Universal Time) format. For example, you could use a command line such as:

```
ldapmodify -p port -h host -D cn=orcladmin -q -v -f my.ldif
```

where my.ldif contains:

```
dn:cn=John Doe,cn=users,o=my_company,dc=com
orclactivestartdate:20030101000000z
orclactiveenddate: 20031231000000z
```

In this example, John Doe can log in only between January 1, 2003 and December 31, 2003. He cannot login before January 1, 2003 or after December 31, 2003. If you want to disable his account for the period between these dates, then set the `orclisenabled` attribute to `DISABLED`.

12.2.2 Unlocking Accounts by Using Command-Line Tools

If you are a member of the Security Administrators Group, then you can unlock an account without resetting the user password. This saves you from having to explicitly tell the user the new password. The user can simply log in using the old password.

To unlock an account, set the `orclpwdaccountunlock` attribute to 1.

The following example unlocks the account for user John Doe.

```
ldapmodify -p port -h host -D cn=orcladmin -q -v -f file.ldif
```

where file.ldif contains:

```
dn: cn=John Doe,cn=users,o=my_company,dc=com
changetype: modify
add: orclpwdaccountunlock
orclpwdaccountunlock: 1
```

12.2.3 Forcing a Password Change by Using Command-Line Tools

You can force users to change their passwords when they log in for the first time. To do this, set the `pwdMustChange` attribute in the `pwdpolicy` entry to `1`, and then reset the password. If

you do this, you must explicitly tell the user the new password so that the user can log in to change that password.

> **See Also:**
>
> - Resetting Your Own Password by Using the Oracle Internet Directory Self-Service Console for instructions on resetting passwords
> - Setting Password Policies by Using Command-Line Tools for instructions on setting attributes of a password policy

12.3 Managing Accounts and Passwords by Using the Self-Service Console

For administrators, Oracle Directory Services Manager is the primary tool for managing users and passwords.

You can also use Oracle Identity Manager to centralize user and account provisioning to Oracle Internet Directory . For end user self-service, Oracle Identity Manager is the recommended solution. The Oracle Identity Manager documentation is available on Oracle Technology Network.

This section contains these topics:

- Enabling and Disabling Accounts by Using the Oracle Internet Directory Self-Service Console
- Unlocking Accounts by Using the Oracle Internet Directory Self-Service Console
- Resetting Your Own Password by Using the Oracle Internet Directory Self-Service Console

12.3.1 Enabling and Disabling Accounts by Using the Oracle Internet Directory Self-Service Console

You can temporarily disable a user's account, then enable it again, by using the Oracle Internet Directory Self-Service Console.

> **See Also:**
>
> Managing Users in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager*.

12.3.2 Unlocking Accounts by Using the Oracle Internet Directory Self-Service Console

If you are a member of the Security Administrators Group, then, if an account becomes locked, you can unlock it without resetting the user password. This saves you from having to explicitly tell the user the new password. The user can simply log in by using the old password.

> **See Also:**
>
> The section on Unlocking a User Account in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager*.

12.3.3 Resetting Your Own Password by Using the Oracle Internet Directory Self-Service Console

If you forget your password or become locked out of your account, then you can reset your password. This involves identifying yourself to the server by providing values for a set of password validation attributes. This takes the form of answering a password hint question to which you had earlier specified an answer.

> **See Also:**
>
> To reset your password using Self-Service console in Oracle Identity Manager, See See Resetting the User Password in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager*.

12.4 Unlocking Locked Accounts by Using Oracle Directory Services Manager

Locked accounts can be listed by using Oracle Directory Services Manager by using the search string `pwdaccountlockedtime`.

To list and unlock locked accounts using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. Perform a simple search, as described in Searching for Entries by Using Oracle Directory Services Manager, using the search string **(pwdaccountlockedtime=*)**. A list of entries with locked accounts appears.
4. Select the entry whose account you want to unlock.
5. When an account is locked, **Unlock Account** appears before the **Apply** and **Revert** buttons. Click **Unlock Account**.

12.5 Changing the Superuser Password by Using Fusion Middleware Control

The configuration attribute `orclsupassword` is an attribute of the DSE root.You can change the super user password which is assigned before.

To change the password for the superuser by using Oracle Enterprise Manager Fusion Middleware Control:

1. Select **Administration**, then **Shared Properties** from the **Oracle Internet Directory** menu.
2. Click the **Change Superuser Password** tab.
3. Specify the old password.
4. Specify the new password.
5. Confirm the new password.
6. Click **Apply**.

12.6 Creating Another Account With Superuser Privileges

The Superuser, `cn=orcladmin`, gets its privileges from membership in several privileged groups.

You can query for those groups by using the following `ldapsearch` command:

```
ldapsearch -h host -p port -D "cn=orcladmin" -q -b "" -L \
-s sub "(|(uniquemember=cn=orcladmin)(member=cn=orcladmin))" dn
```

To create a second account with Superuser privilege, create another user entry that belongs to the same groups. Also add the user as member of the group `cn=directoryadmingroup,cn=oracle internet directory`.

> **Note:**
>
> To use all ODSM features including the Security and Advanced tabs, a new superuser account must be a direct member of the `DirectoryAdminGroup` group. The new superuser account cannot be a member of a group that is in turn a member of the `DirectoryAdminGroup` group. In this configuration, the superuser would be able to access only the ODSM Home, Schema, and Data Browser tabs.

After you have created additional users with Superuser privileges, you no longer need to use `cn=orcladmin` to administer Oracle Internet Directory. The privileged accounts should be sufficient. The attribute `orclsuname`, however, must have the value `cn=orcladmin`.

> **See Also:**
>
> Managing Directory Entries in Oracle Internet Directory to learn how to create a user entry and Managing Dynamic and Static Groups in Oracle Internet Directory to learn how to add a user to a group.

> **Note:**
>
> To maintain system security, keep the number of privileged users to a minimum and ensure that all privileged accounts are audited. See Managing Auditing .

12.7 Managing the Superuser Password by Using ldapmodify

You should never change the Superuser's name. The value of `orclsuname` must remain `cn=orcladmin`.

To set or modify the password for the superuser, use `ldapmodify` to modify the attribute `orclsuname` or `orclsupassword`, respectively, in the DSE root. Changing the user name of the superuser can have serious repercussions and is not recommended.

To change the password of the superuser to `superuserpassword`, use an LDIF file such as the following:

```
dn:
changetype:modify
replace:orclsupassword
orclsupassword:superuserpassword
```

> **See Also:**
>
> The `ldapmodify` command-line tool reference in *Reference for Oracle Identity Management* for `ldapmodify` syntax and usage notes.

12.8 Changing the Oracle Internet Directory Database Password

The Oracle Internet Directory uses a password when connecting to its own designated Oracle database. The default for this password when you install Oracle Internet Directory is the same as that for the Oracle Fusion Middleware administrator. When you change the password using `oidpasswd`, the new password is saved in the wallet. When you try to connect to Oracle Internet Directory's database next time, it will validate the user with the new password saved in the wallet and connect to the database.

You can change this password by using `oidpasswd`.

The following example shows how to change the Oracle Internet Directory database password:

```
oidpasswd connect=OIDDB change_oiddb_pwd=true
current password: oldpassword
new password: newpassword
```

```
confirm password: newpassword
password set.
```

> **See Also:**
>
> The Using oidpasswd command-line tool reference in *Reference for Oracle Identity Management*

> **Note:**
>
> The account described here is different from the ODSSM account used for accessing server manageability information. Account Used for Accessing Server Manageability Information describes that account. For information about changing that account, see Changing the Password for the ODSSM Administrator Account.

12.9 Resetting the Superuser Password

If you forget the Oracle Internet Directory superuser (cn=orcladmin) password, you can use the oidpasswd tool to reset it. You must provide the Oracle Internet Directory database password.

When you first install Oracle Internet Directory, the superuser password and Oracle Internet Directory database password are the same. After installation, however, you can change the Oracle Internet Directory superuser password using ldapmodify. If you forget the Oracle Internet Directory superuser password, you can reset it using the oidpasswd tool separately.

The following example shows how to reset the Oracle Internet Directory superuser password. The oidpasswd tool prompts you for the Oracle Internet Directory database password.

```
oidpasswd connect=OIDDB reset_su_password=true
OID DB user password: oid_db_password
        password: new_su_password
confirm password: new_su_password
OID superuser password reset successfully
```

12.10 Changing the Password for the EMD Administrator Account

The EMD administrator account, "cn=emd admin,cn=oracle internet directory", has very limited privilege and is used primarily for starting and stopping Oracle Internet Directory server manageability information collection.

> **See Also:**
>
> Monitoring Oracle Internet Directory for information about Oracle Internet Directory server manageability information collection.

To change the password for the EMD administrator:

1. Change the `userpassword` of the account `"cn=emd admin,cn=oracle internet directory"` in Oracle Internet Directory by using `ldapmodify` command.
2. Invoke `wlst` command and connect to the WebLogic server.

   ```
   java weblogic.WLST
   connect('weblogic', 'weblogic_user_password', 'protocol:host:port'
   ```

3. Run the `updateCred` WLST command:

   ```
   updateCred(map='emd',key='EMD_instance_name',
   password='newpassword',user='EMD')
   ```

   where *`instance_name`* is the instance name provided during installation, for example, `oid1`.
4. On each Oracle instance in the WebLogic domain, execute the following command line:

   ```
   oid_setProperties(context='EM', host='host', port = nnnn, sslmode=nnn,
   sslwrl = 'file:/wallet-location', emdPassword = 'emd-login-password')
   ```

12.11 Changing the Password for the ODSSM Administrator Account

Oracle Internet Directory connects to its Oracle Database, using the password specified for the ODS schema during schema creation. It also connects to retrieve its metric using the ODSSM schema password, given during schema creation as well. The Oracle Enterprise Manager Fusion Middleware Control default password, at the end of install, is the same as the ODSSM password.

To change the password for the ODSSM administrator, you must change it in the Oracle Database and then change it on both the WebLogic domain server and on each Oracle instance in the domain. Use the following procedure:

1. Use SQLPlus or a similar tool to alter the password in the database.
2. Go to *`ORACLE_HOME`*`/common/bin` and run the following command:

   ```
   sh wlst.sh
   ```

3. Connect to the WebLogic Administration Server:

   ```
   connect('weblogic_username','pwd', 't3://host:port')
   ```

4. Run the `updateCred()` command:

   ```
   updateCred(map='odssm', key='ODSSM_instance_name', password='newpassword',
   user='ODSSM')
   ```

   where *`instance_name`* is the instance name provided during installation, for example, `asinst_1`.
5. On each Oracle instance in the WebLogic domain, execute the following command line:

   ```
   oid_setProperties(context='EM', host='host', port = nnnn, sslmode=nnn,
   sslwrl = 'file:/wallet-location', odssmPassword = 'odssm-schema-password')
   ```

12.12 Updating the New ODSSM Password for Data Source

If Oracle Directory Integration Platform is also configured in the instance, then you must update this new ODSSM password in one additional place.

To update the new ODSSM password in Oracle Internet Directory credential store:

1. Log in to the WebLogic Administration console at: `http://host:port/console`
2. Select **Data Sources** -> **schedulerDS** -> **Connection Pool**.
3. Click **Lock & Edit** in the top left corner of the screen.
4. Enter the new password in the **Password** and **Confirm Password** fields.

   Click **Save**.
5. Click **Activate Change**s.

> **Note:**
>
> You can validate the ODSSM password using the following script:
>
> ```
> %perlbin%/perl $ORACLE_HOME/sysman/admin/scripts/iam/getCSFPassword.pl
> $ORACLE_HOME $ORACLE_INSTANCE [CANONICAL_PATH] ldap
> ```

13

Managing Directory Entries in Oracle Internet Directory

You can manage Oracle Internet Directory directory entries using Oracle Directory Services Manager and LDAP command-line utilities:
The following topics describe managing directory entries:

- Introduction to Managing Directory Entries
- Managing Entries by Using Oracle Directory Services Manager
- Managing Entries by Using LDAP Command-Line Tools

13.1 Introduction to Managing Directory Entries

The primary function of most directories is to store information about users and return that information in response to requests. Applications that request information from the directory server are called clients of the server.

As administrator, you manage users, groups, and other types of entries by using Oracle Directory Services manager or the command-line tools.

> **See Also:**
>
> Understanding the Concepts and Architecture of Oracle Internet Directory, for introductory information about entries, object classes, and attributes.

13.2 Managing Entries by Using Oracle Directory Services Manager

You display entries, including users and groups, by using the Data Browser in Oracle Directory Services Manager.

The current chapter focuses on users and other types of entries. Managing Dynamic and Static Groups in Oracle Internet Directory discusses groups and group entries in more detail.

This section contains these topics:

- Displaying Entries by Using Oracle Directory Services Manager
- Searching for Entries by Using Oracle Directory Services Manager
- Importing Entries from an LDIF File by Using Oracle Directory Services Manager
- Exporting Entries to an LDIF File by Using Oracle Directory Services Manager
- Viewing Attributes for a Specific Entry by Using Oracle Directory Services Manager
- Adding a New Entry by Using Oracle Directory Services Manager

- Deleting an Entry or Subtree by Using Oracle Directory Services Manager
- Adding an Entry by Copying an Existing Entry in Oracle Directory Services Manager
- Modifying an Entry by Using Oracle Directory Services Manager

> **See Also:**
>
> - Adding or Modifying an ACP by Using the Data Browser in ODSM
> - Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM
>
> for information on setting or modifying access control on an entry.

13.2.1 Displaying Entries by Using Oracle Directory Services Manager

Entries of some object class types have generic icons in the data tree.

Other object entries are shown with a specific icon. For example:

Object Class	Icon
User	
Group	
OrganizationalUnit	
Organization	
Domain	
Country	

Object Class	Icon
Generic	

When an access control list (ACL) has been set on an entry, the icon changes; a small key appears to the right of the icon. For example:

Object Class	Icon with ACL
User	
Group	

> **See Also:**
>
> Managing Directory Access Control.

To display entries by using the Data Browser in Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. If desired, expand items in the data tree in the left panel to view the entries in each subtree.
4. If desired, mouse over each icon in the tool bar to read the icon's action.
5. Select the **Refresh the entry** icon to refresh only the entry in the right pane. Select the **Refresh subtree entries** icon to refresh child entries of the selected entry.
6. To limit the number of entries displayed in a subtree, select the entry at the root of the subtree, then click the **Filter child entries** icon and specify a filter, as follows:
   a. In the Max Results field, specify a number from 1 to 1000, indicating the maximum number of entries to return.
   b. From the list at the left end of the search criteria bar, select an attribute of the entries you want to view.
   c. From the list in the middle of the search criteria bar, select a filter.

d. In the text box at the right end of the search criteria bar, type the value for the attribute you just selected. For example, if the attribute you selected was `cn`, you could type the particular common name you want to find.

e. Click **+** to add this search criterion to the **LDAP Query** field.

f. To view the LDAP filter you have selected, select **Show LDAP filter**.

g. To further refine your search, use the list of conjunctions (**AND**, **OR**, **NOT AND**, and **NOT OR**) and the lists and text fields on the search criteria bar to add additional search criteria. Click **+** to add a search criterion to the **LDAP Query** field. Click **X** to delete a search criterion from the **LDAP Query** field.

7. When you have finished configuring the search criteria, click **OK**. The child entries that match the filter are shown under the selected entry. The filter is applied for first level children only, not for the entire subtree. Click the **Refresh** icon to remove the filter.

13.2.2 Searching for Entries by Using Oracle Directory Services Manager

You can invoke simple and advanced search for entries using Oracle Directory Services Manager.

To search for a directory entry:

1. Invoke Oracle Directory Services Manager as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Data Browser**.
3. To perform a simple keyword search, enter text in the field next to the Search icon to specify keywords to search for in the attributes `cn`, `uid`, `sn`, `givenname`, `mail` and `initials`.
4. Click the **Simple Search** arrow to the right of the text field or press the Enter key. Search results, if any, are displayed below the data tree. Click the information icon to view information about this search. Click the **Refresh the search results entries** icon to refresh the results. Click the **Close search result** icon to dismiss the search.
5. To perform a more complex search, click **Advanced**. The Search Dialog appears.
6. In the **Root of the Search** field, enter the DN of the root of your search.

   For example, suppose you want to search for an employee who works in the Manufacturing division in the IMC organization in the Americas. The DN of the root of your search would be:

   ```
   ou=Manufacturing,ou=Americas,o=IMC,c=US
   ```

   You would therefore type that DN in the **Root of the Search** text box.

   You can also select the root of your search by browsing the data tree. To do this:

   a. Click **Browse** to the right of the **Root of the Search** field. The Select Distinguished Name (DN) Path: Tree View dialog box appears.

   b. Expand an item in the tree view to display its entries.

   c. Continue navigating to the entry that represents the level you want for the root of your search.

   d. Select that entry, then click **OK**. The DN for the root of your search appears in the **Root of the Search** text box in the right pane.

7. In the **Max Results (entries)** box, type the maximum number of entries you want your search to retrieve. The default is 200. The directory server retrieves the value you set, up to 1000.

8. In the **Max Search Time (seconds)** box, type the maximum number of seconds for the duration of your search. The value you enter here must be at least that of the default, namely, 25. The directory server searches for the amount of time you specify, up to one hour.
9. In the **Search Depth** list, select the level in the DIT to which you want to search.

   The options are:

   - **Base**: Retrieves a particular directory entry. Along with this search depth, you use the search criteria bar to select the attribute `objectClass` and the filter `Present`.
   - **One Level**: Limits your search to all entries beginning one level down from the root of your search.
   - **Subtree**: Searches entries within the entire subtree, including the root of your search. This is the default.
10. Set search criteria.

    Optionally, select Show LDAP filter, then type a query string directly into the **LDAP Query** text field.

    Alternatively, use the lists and text fields on the search criteria bar to focus your search.

    a. From the list at the left end of the search criteria bar, select an attribute of the entry for which you want to search. Because not all attributes are used in every entry, be sure that the attribute you specify actually corresponds to one in the entry for which you are looking. Otherwise, the search fails.

    b. From the list in the middle of the search criteria bar, select a filter.

    c. In the text box at the right end of the search criteria bar, type the value for the attribute you just selected. For example, if the attribute you selected was `cn`, you could type the particular common name you want to find.

    d. Click **+** to add this search criterion to the **LDAP Query** field.

    e. To view the LDAP filter you have selected, select **Show LDAP filter**.

    f. To further refine your search, use the list of conjunctions (**AND**, **OR**, **NOT AND**, and **NOT OR**) and the lists and text fields on the search criteria bar to to add additional search criteria. Click **+** to add a search criterion to the **LDAP Query** field. Click **X** to delete a search criterion from the **LDAP Query** field.
11. Click **Search**. Search results, if any, are displayed below the data tree. If an **LDAP error** icon appears, mouse over it to see the error. Search again with different criteria, if necessary, to correct the error. Click the Search Filter icon to see information about the search. Chick the **Refresh the search result entries** icon to refresh the results. You can delete the search results by clicking the **Close search result** icon.

> **See Also:**
>
> Viewing Active Server Instance Information by Using WLST Command — oid_instanceStatus() For instructions on setting the number of entries to display in searches, and to set the time limit for searches

13.2.3 Importing Entries from an LDIF File by Using Oracle Directory Services Manager

You can import entries from an LDIF file using Oracle Directory Services Manager.

To import entries from an LDIF file:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. Click the Data Browser tab.
3. Click the **Import LDIF** icon. The Import File dialog appears.
4. Enter the path to the LDIF file you want to import, or click **Browse** and navigate to the file, then click **Open** in the browser window.
5. Click **OK** in the Import File dialog. The LDIF Import Progress window shows the progress of the operation. Expand View Import Progress Table to see detailed progress.

   Click **Cancel** to stop importing entries. Entries already imported are not aborted.

   The Data Browser tree refreshes to show the new entries.

13.2.4 Exporting Entries to an LDIF File by Using Oracle Directory Services Manager

You can export entries to an LDIF file by using Oracle Directory Services Manager.

To export entries to an LDIF file:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. Click the Data Browser tab.
3. Navigate to the top level DN of the subtree you want to export.
4. Click the **Export LDIF** icon. The Export File dialog appears. Select Export Operational Attributes if you want to export them.
5. Click **OK**. The Download LDIF File dialog appears. By default, the entries are exported to a temporary file on the machine where Oracle Directory Services Manager is deployed. If you want to save a copy of the LDIF file to your computer, click **Click here to open the LDIF file** and save the file.

   Click **OK**.

13.2.5 Viewing Attributes for a Specific Entry by Using Oracle Directory Services Manager

You can view attributes for a specific entry by using Oracle Directory Services Manager.

To view the attributes for a specific entry:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.

2. Locate the entry by navigating to it in the data tree or by searching for it, as described in Searching for Entries by Using Oracle Directory Services Manager.
3. Click the entry. Attributes for that entry are displayed in the right pane. The display for the entry has at least the three tabs: **Attributes**, **Subtree Access**, and **Local Access**. If the entry is a person, the display in the right pane also has an **Person** tab, which displays basic user information. If the entry is a group, the display screen has a **Group** tab, which displays basic group information.
4. To view the attributes of an entry, click the **Attributes** tab.
5. You can switch between Managed Attributes and Show All by using the **Views** list.
6. To change the list of attributes shown as managed attributes, click the icon under **Optional Attributes**. Select attributes you want to move from the All Attributes list to the Shown Attributes lists and use the **Move** and **Move All** arrows to move the attributes. Select attributes you want to move from the shown Attributes list to the All Attributes lists and use the **Remove** and **Remove All** arrows to move the attributes. Click **Add Attributes** to make your changes take effect or click **Cancel** to discard your changes. After you click **Add Attributes**, only the attributes that were on the Shown Attributes list are shown in the Managed Attributes view.

For information on using the Subtree Access and Local Access tabs to view access control settings, see Adding or Modifying an ACP by Using the Data Browser in ODSM.

13.2.6 Adding a New Entry by Using Oracle Directory Services Manager

To add or delete entries with Oracle Directory Services Manager, you must have write access to the parent entry and you must know the DN to use for the new entry.

> **Note:**
>
> When you add or modify an entry, the Oracle directory server does not verify the syntax of the attribute values in the entry.

To add a group entry, follow the procedure described in Managing Group Entries by Using Oracle Directory Services Manager.

To add an entry other than a group entry type:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. On the toolbar, select the **Create a new entry** icon. Alternatively, right click any entry and choose **Create**.

   The Create New Entry wizard appears.
4. Specify the object classes for the new entry. Click the **Add** icon and use the Add Object Class dialog to select object class entries. Optionally, use the search box to filter the list of object classes. To add the object class, select it and then click **OK**. (All the superclasses from this object class through `top` are also added.)
5. In the **Parent of the entry** field, you can specify the full DN of the parent entry of the entry you are creating. You can also click **Browse** to locate and select the DN of the parent for the entry you want to add. If you leave the **Parent of the entry** field blank, the entry is created under the root entry.

6. Click **Next**.
7. Choose an attribute which will be the **Relative Distinguished Name** value for this entry and enter a value for that attribute. You must enter values for attributes that are required for the object class you are using, even if none of them is the RDN value. For example, for object class `inetorgperson`, attributes `cn` (common name) and `sn` (surname or last name) are required, even if neither of them is the Relative Distinguished Name value.
8. Click **Next**. The next page of the wizard appears. (Alternatively, you can click **Back** to return to the previous page.)
9. Click **Finish**.

13.2.7 Deleting an Entry or Subtree by Using Oracle Directory Services Manager

You can delete an entry or subtree by using Oracle Directory Services Manager.

To delete an entry, including an entire subtree:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. Navigate to the entry you want to delete.
4. To delete only the entry, click the **Delete** icon. When the Delete dialog appears, click **Yes**. If the entry has no subentries, deletion succeeds. If the entry has subentries, the deletion fails and ODSM displays an error message. Click **OK** to dismiss the error message.

   To delete an entire subtree, click the icon labelled **Delete the selected entry and its subtree.** When the Delete Subtree dialog appears, read the contents of the dialog. Click **Yes** to proceed with the deletion or **No** to abort.

> **Note:**
>
> Before you delete an entire subtree with a large number of entries, configure the undo tablespace size so that it has sufficient space for the delete operation.
>
> For more information, see Managing Undo in *Oracle Database Administrator's Guide*.

13.2.8 Adding an Entry by Copying an Existing Entry in Oracle Directory Services Manager

You can use Oracle Directory Services Manager to create a new entry by copying from an existing entry and changing its DN. When you do this, you should also change the attributes, such as name and address, so that they correspond with the new DN.

To add an entry, you must have write access to its parent.

> **Tip:**
>
> You can find a template for the new DN by looking up other similar entries in the search pane.

To add a group entry, follow the procedure described in Managing Group Entries by Using Oracle Directory Services Manager.

To add an entry (other than a group entry type) by copying an existing entry:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. In the data tree, navigate to the entry you want to use as a template. Alternatively, click **Advanced Search**, and use it to search for an entry that you want to use as a template.
4. In the left panel, click the **Create a new entry like this one** icon. Alternatively, click the entry you want to use as a template, right click, and choose **Create Like**. A **New Entry: Create Like** wizard appears. The object classes and the DN of the parent entry are already filled in.
5. To add an object class:
   a. Click the **Attributes** tab.
   b. Click the **Add** icon next to `objectclass` and use the Add Object Class dialog to select object class entries. Optionally, use the search box to filter the list of object classes. To add the object class, click it and then click **OK**.
6. To delete an object class,
   a. Click the **Attributes** tab.
   b. Select the object class you want to delete.
   c. Click the **Delete** icon next to `objectclass`. The Delete Object Class dialog lists the attributes that will be deleted with that class.
   d. Click **Delete** to proceed.
7. Specify the DN of the parent entry, either by changing the content in the text box or by using the **Browse** button to locate a different DN.
8. Click **Next**. The next page of the wizard appears.
9. Choose an attribute which will be the **Relative Distinguished Name** value for this entry and enter a value for that attribute. You must enter values for attributes that are required for the object class you are using, even if none of them is the RDN value. For example, for object class `inetorgperson`, attributes `cn` (common name) and `sn` (surname or last name) are required, even if neither of them is the Relative Distinguished Name value.
10. Click **Next**.
11. Click **Finish**.

13.2.9 Modifying an Entry by Using Oracle Directory Services Manager

You can add auxiliary object classes to an existing entry.

> **Note:**
>
> When you add or modify an entry, the Oracle directory server does not verify the syntax of the attribute values in the entry.

To modify a group entry, follow the procedure described in Managing Group Entries by Using Oracle Directory Services Manager. For other entry types, proceed as follows:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. Navigate to an entry in the data tree. Alternatively, perform a search for the entry you want to modify as described in Searching for Entries by Using Oracle Directory Services Manager. In the search result in the left pane, select the entry you want to modify.
4. To edit the RDN, select the **Edit RDN** icon above the Data Tree. Alternatively, you can select the entry in the Data Tree, right click, and select **Edit RDN**.

   Specify the new RDN value. For a multivalued RDN you can use the **Delete Old RDN** checkbox to specify whether the old RDN should be deleted. Select **OK** to save the change or **Cancel** to abandon the change.
5. To add an object class:
   a. Click the **Attributes** tab.
   b. Click the **Add** icon next to `objectclass` and use the Add Object Class dialog to select object class entries. Optionally, use the search box to filter the list of object classes. To add the object class, click it and then click **OK**.
6. To delete an object class,
   a. Click the **Attributes** tab.
   b. Select the object class you want to delete.
   c. Click the **Delete** icon next to `objectclass`. The Delete Object Class dialog lists the attributes that will be deleted with that class.
   d. Click **Delete** to proceed or **Cancel** to cancel the deletion.
7. If the entry is a person, click the **Person** tab and use it to manage basic user attributes. Click **Apply** to save your changes or **Revert** to discard them.

   If the entry is a group, see Managing Group Entries by Using Oracle Directory Services Manager.
8. If this is a person entry, you can upload a photograph. Click **Browse**, navigate to the photograph, then click **Open**. To update the photograph, click **Update** and follow the same procedure. Click the **Delete** icon to delete the photograph.
9. To modify the values of attributes that are not specific to a person or group, click the **Attributes** tab in the right pane and make the desired changes.

By default, only non-empty attributes are shown. You can switch between **Managed Attributes** and **Show All** by using the **Views** list.

10. To change the list of attributes shown as managed attributes, click the icon under **Optional Attributes**. Select attributes you want to move from the All Attributes list to the Shown Attributes lists and use the **Move** and **Move All** arrows to move the attributes. Select attributes you want to move from the shown Attributes list to the All Attributes lists and use the **Remove** and **Remove All** arrows to move the attributes. Click **Add Attributes** to make your changes take effect or click **Cancel** to discard your changes. After you click **Add Attributes**, only the attributes that were on the Shown Attributes list are shown in the Managed Attributes view.
11. Specify values for the optional properties. You can also modify the values of the mandatory properties. For multivalued attributes, you can use the **Add** and **Delete** icons to add and delete multiple values.
12. When you have completed all your changes, click **Apply** to make them take effect. Alternatively, click **Revert** to abandon your changes.
13. You can set an access control point (ACP) on this entry by using the Subtree Access and Local Access tabs. The procedures are described in Adding or Modifying an ACP by Using the Data Browser in ODSM and Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM.

13.3 Managing Entries by Using LDAP Command-Line Tools

You can manage entries using LDAP Command-Line tools.

This section contains the following topics:

- Listing All the Attributes in the Directory by Using ldapsearch
- Listing Operational Attributes by Using ldapsearch
- Changing the Attribute Case in `ldapsearch` Output
- Adding a User Entry by Using ldapadd
- Modifying a User Entry by Using ldapmodify
- Adding an Attribute Option by Using ldapmodify
- Deleting an Attribute Option by Using ldapmodify
- Searching for Entries with Attribute Options by Using ldapsearch

13.3.1 Listing All the Attributes in the Directory by Using ldapsearch

`ldapsearch` command is used in listing the attributes in the directory.

Use the following command line to list of all the attributes, including those that do not have values:

```
ldapsearch -p port -h host -D "cn=orcladmin" -q -b "cn=subschemasubentry" \
           -s base "objectclass=*"
```

13.3.2 Listing Operational Attributes by Using ldapsearch

By default, `ldapsearch` does not return operational attributes. If you add the character "**+**" to the list of attributes in the search request, however, `ldapsearch` returns all operational attributes.

Searching for an entry with "**+**" returns only operational attributes. For example:

```
$ ldapsearch -h example.com -p 3060 -D cn=orcladmin -w password -b "c=uk" -L -s base
"(objectclass=*)" +
dn: c=UK
orclguid: 8EB5730F5852DECBE040E80A7452694E
creatorsname: cn=orcladmin
createtimestamp: 20100826065339z
modifytimestamp: 20100826065339z
modifiersname: cn=orcladmin
orclnormdn: c=uk
```

By comparison, a search with "*****" but not "**+**" returns all user attributes:

```
$ ldapsearch -h example.com -p 3060 -D cn=orcladmin -w password -b "c=uk" -L -s base
"(objectclass=*)"
dn: c=UK
c: uk
objectclass: top
objectclass: country
```

13.3.3 Changing the Attribute Case in `ldapsearch` Output

In the output from the `ldapsearch` command, the attribute names are shown in lower case if the attribute `orclReqattrCase`in the instance-specific configuration entry is `0`. If `orclReqattrCase` is set to `1`, the attribute names in the output are shown in the same case in which they were entered on the command line.

Example:

```
ldapsearch -h localhost -p 3060 -b "dc=oracle,dc=com" -s base -L "objectclass=*" DC
```

If `orclReqattrCase` is `0` the output looks like this:

```
dn: dc=oracle,dc=com
dc: oracle
```

If `orclReqattrCase` is `1`, the output looks like this:

```
dn: dc=oracle,dc=com
DC: oracle
```

If an attribute is specified more than once on the same command line, the attribute names in the output will match the case of the first attribute specification.

13.3.4 Adding a User Entry by Using ldapadd

`ldapadd` command is used to add a user entry.

The following example shows how to add an entry for an employee named John.

Use `ldapadd` as follows:

```
ldapadd -p port_number -h host -D cn=orcladmin -b -q -f entry.ldif
```

where entry.ldif looks like this:

```
dn: cn=john, c=us
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: john
cn;lang-fr:Jean
cn;lang-en-us:John
sn: Doe
jpegPhoto: /photo/john.jpg
userpassword: password
```

This file contains the `cn`, `sn`, `jpegPhoto`, and `userpassword` attributes.

For the `cn` attribute, it specifies two options: `cn;lang-fr`, and `cn;lang-en-us`. These options return the common name in either French or American English.

For the `jpegPhoto` attribute, it specifies the path and file name of the corresponding JPEG image you want to include as an entry attribute.

> **Note:**
>
> - When you add or modify an entry, the Oracle directory server does not verify the syntax of the attribute values in the entry.
> - Do not insert a tilde (~) in a user name.

13.3.5 Modifying a User Entry by Using ldapmodify

`ldapmodify` command modifies a user entry.

The following example changes the password for a user to a new value. As in the previous example, the data for this user entry is in the `entry.ldif` file. This file contains the following:

```
dn: cn=audrey,c=us
changetype: modify
replace: userpassword
userpassword: password
```

Substitute the new password for *password* in the file.

Issue this command to modify the file:

```
ldapmodify -p 3060 -D "cn=orcladmin" -q -v -f entry.ldif
```

where -v specifies verbose mode.

> **Note:**
>
> When you add or modify an entry, the Oracle directory server does not verify the syntax of the attribute values in the entry.

13.3.6 Adding an Attribute Option by Using ldapmodify

`ldapmodify` command is used to modify a file.

The following entry adds the Spanish equivalent of an entry for John. The data for this user entry is in the `entry.ldif` file. This file contains the following:

```
dn: cn=john,c=us
changetype: modify
add: cn;lang-sp
cn;lang-sp: Juan
```

Issue this command to modify the file:

```
ldapmodify -D "cn=orcladmin" -q -p 3060 -v -f entry.ldif
```

13.3.7 Deleting an Attribute Option by Using ldapmodify

You can delete an attribute entry by `ldapmodify` command.

The following example deletes the `cn;lang-fr` attribute option from the entry for John. As in the previous example, assume that the data for this user entry is in the `entry.ldif` file. This file contains the following:

```
dn: cn=john, c=us
changetype: modify
delete: cn;lang-fr
cn;lang-fr: Jean
```

Issue this command to modify the file:

```
ldapmodify -D "cn=orcladmin" -q -p 3060 -v -f entry.ldif
```

13.3.8 Searching for Entries with Attribute Options by Using ldapsearch

`ldapsearch` is used to search for entries with attribute options.

The following example retrieves entries with common name (`cn`) attributes that have an option specifying a language code attribute option. This particular example retrieves entries in which the common names are in French and begin with the letter R.

```
ldapsearch -D "cn=orcladmin" -q -p 3060 -h myhost -b "c=US" -s sub "cn;lang-fr=R*"
```

Suppose that, in the entry for John, no value is set for the `cn;lang-it` language code attribute option. In this case, the following example fails:

```
ldapsearch -D "cn=orcladmin" -q -p 3060 -h myhost -b "c=us" \
           -s sub "cn;lang-it=Giovanni"
```

> **See Also:**
>
> Attribute Options.

You can use the `-X` or `-B` options to `ldapsearch` to print binary values.

> **See Also:**
>
> The `ldapsearch` command reference in *Reference for Oracle Identity Management*.

14

Managing Dynamic and Static Groups in Oracle Internet Directory

You can manage both static and dynamic groups in Oracle Internet Directory using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities.

- Understanding Dynamic and Static Groups
- Managing Group Entries by Using Oracle Directory Services Manager
- Managing Group Entries by Using the Command Line

14.1 Understanding Dynamic and Static Groups

Oracle Internet Directory enables you to assign and manage membership in two types of groups—namely, static groups and dynamic groups. Each type of group suited for a different purpose.

> **Note:**
>
> If you are creating a hierarchy of groups, be sure that it is a true hierarchy as described in About Hierarchies of Group Entries.

> **See Also:**
>
> - About Security Groups for instructions on setting access control policies for group entries
> - Globalization Support and Managing Directory Access Control for information about access privileges.

This section contains these topics:

- Defining Static Groups
- Defining Dynamic Groups
- About Hierarchies of Group Entries
- About Querying Group Entries
- Understanding the `orclMemberOf` Attribute
- Considerations for Using Static and Dynamic Group

14.1.1 Defining Static Groups

Contextual description of static groups and the schema elements required to a create a static group.

- Static Group
- Schema Elements for Creating a Static Group

14.1.1.1 Static Group

A static group is one whose entry contains a list of members that you explicitly administer.

A static group requires you to explicitly administer its membership. For example, if a member changes his name, then you must change that user's DN for each group he belongs to. For this reason, a static group is best suited for a group whose membership is unlikely to change frequently.

14.1.1.2 Schema Elements for Creating a Static Group

When you create the entry for this kind of group, you associate it with either the `groupOfNames` or `groupOfUniqueNames` object class.

Each of these object classes has a multivalued attribute for storing the names of group members. To assign a user as a member of a group, you add the DN of each member to the respective multivalued attribute. Conversely, to remove a member from a group, you delete the member's DN from the respective attribute. In the `groupOfNames` object class, this multivalued attribute is `member`, and, in the `groupOfUniqueNames` object class, it is `uniqueMember`.

14.1.2 Defining Dynamic Groups

Contextual description of dynamic group and the schema elements is required to create a dynamic group.

- Dynamic Group
- Cached and Uncached Dynamic Groups
- Enhancements of Dynamic Groups in Oracle Internet Directory
- Limitations of Dynamic Groups in Oracle Internet Directory
- Schema Elements for Creating a Dynamic Group
- About labeledURI Attribute
- About `CONNECT BY` Assertion
- Example of a Dynamic Group Entry Using the labeledURI Attribute
- Example of a Dynamic List Entry Using the labeledURI Attribute
- Example of a Dynamic Group Entry Using the CONNECT BY Assertion

14.1.2.1 Dynamic Group

A dynamic group is one whose membership, rather than being maintained in a list, is computed, based on rules and assertions you specify. Oracle Internet Directory supports the following methods for dynamically computing the membership of the group:

- Using `orclDynamicGroup` object class and `labeleduri` attribute
- Using `orclDynamicGroup` object class and `CONNECT_BY` attributes
- Using `orclDynamicList` object class and `labeleduri` attribute (referred as dynamic list)

Dynamic groups can have static and dynamic members. The static members are listed as values of the `member` or `uniquemember` attribute.

14.1.2.2 Cached and Uncached Dynamic Groups

Dynamic groups can be cached or uncached. By cached, we mean that dynamic group members are computed and stored when the dynamic group is added, and that the member list is kept consistent when the dynamic group is later modified. As entries are added, modified, deleted, and renamed, the member lists of all dynamic groups are kept consistent. For example, if there is a dynamic group containing all `person` entries under "`c=us`", when we add "`cn=user1,c=us`", that entry is automatically added to the member list of the dynamic group. Similarly, when we delete "`cn=user1,c=us`", the entry is removed from the dynamic group's member list. This feature ensures that whenever a search is performed for a dynamic group, the member list can be fetched from the stored data without any additional computation. The search performance for cached dynamic groups is almost the same as for static groups.

Cached Dynamic Group

Starting with Oracle Internet Directory 10g (10.1.4.0.1), dynamic groups based on `orclDynamicGroup` object class using `labeleduri` attribute are cached.

Uncached Dynamic Group

- Dynamic groups based on `orclDynamicGroup` object class using `CONNECT_BY` attributes are not cached.
- Since Oracle Internet Directory 11g Release 1 (11.1.1.4.0), a second type of dynamic group based on `labeleduri` attribute is available. It is referred to as a dynamic list, and its members are not cached. You determine whether a dynamic group based on the `labeleduri` attribute is cached or uncached by selecting the type of auxiliary object class your group is associated with, as described in Schema Elements for Creating a Dynamic Group. If you want a cached group, associate your group with the auxiliary object class `orclDynamicGroup`. If you want an uncached group, associate your group with the auxiliary object class or `orclDynamicList` object class.

Note:

- You cannot add a dynamic group based on the `labeledURI` attribute with scope `base`. Only scope `sub` and `one` are supported.
- To refresh dynamic group memberships for dynamic groups using the `orclDynamicGroup` object class and `labeleduri` attribute, set the attribute `orclrefreshdgrmems` in the DSA Configuration entry to `1`. Oracle Internet Directory recomputes the member lists for all dynamic groups and resets the value of `orclrefreshdgrmems` to `0`. If there are many groups, this operation can take a long time to complete.
- When you query for the groups that a user belongs to, dynamic groups based on the `labeledURI` attribute are automatically included in the result. Dynamic groups based on the `CONNECT_BY` assertion and dynamic lists must be explicitly queried. For example, assume `nc=jdoe,cn=users,o=oracle` is a member of three groups: `labeleduri` dynamic group `dgrouplab1`, `CONNECT BY` dynamic group `dgroupcby1`, and dynamic list `dlist1`. The search `uniquemember=cn=jdoe,cn=users,o=oracle` finds only the cached `labeleduri` dynamic group `dgrouplab1`.

See Also:

- About LDAP Controls in *Reference for Oracle Identity Management* for more information on controls used by Oracle Internet Directory
- Overview of Oracle Internet Directory C API*Application Developer's Guide for Oracle Identity Management*
- Performing Hierarchical Searches Using CONNECT_BY Control*Application Developer's Guide for Oracle Identity Management*

14.1.2.3 Enhancements of Dynamic Groups in Oracle Internet Directory

In Oracle Internet Directory 10g (10.1.4.1) and later releases, you can use dynamic groups in the same ways you use static groups. For example, you can use them in:

- Access control lists, by associating the group with either the `orclACPgroup` or the `orclPrivilegeGroup` object class.
- Hierarchical group resolution queries

14.1.2.4 Limitations of Dynamic Groups in Oracle Internet Directory

Dynamic groups have the following limitations in Oracle Internet Directory:

- Hierarchical queries and queries involving specific attributes of members can only be done on cached dynamic groups.
- Dynamic groups can only be added using `ldapadd` or ODSM. They cannot be added by using `bulkload`.
- The attributes used in the LDAP filter part of the `labeleduri` must be indexed. See Indexing an Attribute by Using ldapmodify, Creating and Dropping Indexes from Existing

Attributes by Using catalog, and Index option in Oracle Internet Directory to Search Attributes .

- You cannot change the objectclass of a dynamic group after the group has been created. You must delete the group and re-create it.
- Searches for the `uniquemember` attribute will not pick up dynamic lists or `CONNECT BY` assertion-based dynamic groups.

14.1.2.5 Schema Elements for Creating a Dynamic Group

When you create a dynamic group, you begin as when creating a static group—that is, you associate its entry with either the `groupOfNames` or `groupOfUniqueNames` object class. You then associate that object class with the auxiliary object class `orclDynamicGroup` or `orclDynamicList`.

The auxiliary object class `orclDynamicGroup` has various attributes in which you specify one of two methods for dynamically computing the membership of the group: using the `labeledURI` attribute and using a `CONNECT BY` assertion. The auxiliary object class `orclDynamicList` supports only the `labeledURI` attribute method of computing membership.

14.1.2.6 About labeledURI Attribute

Both of the auxiliary object classes `orclDynamicGroup` and `orclDynamicList` have the `labeledURI` attribute. If you associate your group with `orclDynamicGroup` and use the `labeledURI` attribute to compute membership, the group is cached. If you associate your group with `orclDynamicList` and use the `labeledURI` attribute to compute membership, the group is not cached. This uncached type, using `orclDynamicList` objectclass, is referred to as a dynamic list.

When using the `labeledURI` method, the directory server performs a typical search based on the hierarchy of the DIT. It requires you to provide a value for one of the attributes of the `orclDynamicGroup` or `orclDynamicList` object class, namely `labeledURI`. In this attribute, you specify the base of the query, the filters, and any required attributes. For example, suppose that you have entered the following value for the `labeledURI` attribute:

```
labeledURI:ldap://host:port/ou=NewUnit,o=MyCompany,c=US??sub?(objectclass=person)
```

When you use this method, a search for the entry returns entries for all members of the group.

Do not set `orclConnectByAttribute` or `orclConnectByStartingValue` when using the `labeledURI` attribute method.

> **Note:**
>
> In the labeledURI attribute, the *host*:*port* section is present for syntax purposes alone. Irrespective of the host and port settings in the labeledURI attribute, the directory server always computes members of dynamic group from the local directory server. It cannot retrieve members from other directory servers.

> **See Also:**
>
> "The LDAP URL Format" (RFC 2255). T. Howes, M. Smith, December 1997. This RFC provides more information about how LDAP URLs are to be represented—as, for example, in the `labeledURI` attribute. It is available at `http://www.ietf.org`.

14.1.2.7 About `CONNECT BY` Assertion

Unlike the `labeledURI` attribute method, this method relies not on the hierarchy of the DIT, but on attributes that implicitly connect entries to each other, regardless of their location in the DIT. For example, the `manager` attribute connects the entries of employees with those of their managers, and this connection applies regardless of the location of the employee entries in the DIT. This method uses a `CONNECT BY` clause in which you specify the attribute to use for building the hierarchy—for example, `manager`—and the starting value for such a hierarchy—for example, `cn=Anne Smith,cn=users,dc=example,dc=com`.

> **See Also:**
>
> Performing Hierarchical Searches in *Application Developer's Guide for Oracle Identity Management*.

More specifically, to use this method, you specify in the `orclDynamicGroup` object class a value for each of the single-valued attributes in Table 14-1.

Table 14-1 orclDynamicGroup Attributes for "Connect By" Assertions

Attribute	Description
orclConnectByAttribute	The attribute that you want to use as the filter for the query—for example, `manager`. This attributed must be indexed.
orclConnectByStartingValue	The DN of the attribute you specified in the `orclConnectByAttribute` attribute—for example, `cn=Anne Smith,cn=users,dc=example,dc=com`

For example, to retrieve the entries of all employees who report to Anne Smith in the MyOrganizational Unit in the Americas, you would provide values for these attributes as follows:

```
orclConnectByAttribute=manager
orclConnectByStartingValue= "cn=Anne Smith,ou=MyOrganizationalUnit,o=MyCompany,c=US"
```

Do not set `labeledURI` when using the `CONNECT BY` assertion method.

You can also develop an application specifying that you want the values for a particular attribute—for example, the `email` attribute—of all the members.

> **See Also:**
>
> Developing Applications for Oracle Identity Management in *Application Developer's Guide for Oracle Identity Management* for more information about how to develop applications that retrieve values for particular attributes.

14.1.2.8 Example of a Dynamic Group Entry Using the labeledURI Attribute

The following is an example of a dynamic group entry using the `labeledURI` attribute.

```
dn: cn=dgroup1
cn: dgroup1
description: this is an example of a dynamic group
labelеduri:ldap://hostname:7777/ou=oid,l=amer,dc=oracle,
 dc=dgrptest??sub?objectclass=person
objectclass: orcldynamicgroup
objectclass: groupOfUniqueNames
objectclass: top
```

This group will have `uniquemember` values that are the DNs of all entries associated with the object class `person` in the subtree `ou=oid,l=amer,dc=oracle,dc=dgrptest`.

14.1.2.9 Example of a Dynamic List Entry Using the labeledURI Attribute

The following is an example of a dynamic list entry using the `labeledURI` attribute. (Dynamic lists are not cached.) It is the same as the previous example, except that the auxiliary object class is `orclDynamicList` instead of `orclDynamicGroup`

```
dn: cn=dgroup1
cn: dgroup1
description: this is an example of a dynamic group
labelеduri:ldap://hostname:7777/ou=oid,l=amer,dc=oracle,
 dc=dgrptest??sub?objectclass=person
objectclass: orcldynamiclist
objectclass: groupOfUniqueNames
objectclass: top
```

This group will have `uniquemember` values that are the DNs of all entries associated with the object class `person` in the subtree `ou=oid,l=amer,dc=oracle,dc=dgrptest`. Searches for the `uniquemember` attribute, however, will not pick up dynamic lists

14.1.2.10 Example of a Dynamic Group Entry Using the CONNECT BY Assertion

The following is an example of a dynamic group entry that uses the CONNECT_BY assertion.

```
dn: cn=dgroup2
cn: dgroup2
description: this is connect by manager assertion dynamic group
orclconnectbyattribute: manager
orclconnectbystartingvalue: cn=john doe sr,l=amer,dc=oracle,dc=dgrptest
objectclass: orcldynamicgroup
objectclass: groupOfUniqueNames
objectclass: top
```

This dynamic group has unique members with values that are DNs of all the entries whose `manager` attribute is `cn=john doe sr`. either indirectly or directly. If several individuals have

`cn=john doe JR`. as their manager, and he, in turn, has `cn=john doe SR`. as his manager, then all the lower-level individuals are returned.

14.1.3 About Hierarchies of Group Entries

Hierarchies can be either explicit or implicit. In explicit hierarchies, the relationship is determined by the location of the entry in the DIT—for example, Group A may reside higher in the DIT than Group B.

In implicit hierarchies, the relationship between entries is determined not by the location in the DIT, but by the values of certain attributes. For example, suppose that you have a DIT in which the entry for John Doe is at the same level of the hierarchy as Anne Smith. However, suppose that, in the entry for John Doe, the `manager` attribute specifies Anne Smith as his manager. In this case, although their locations in the DIT are at an equal level, their rankings in the hierarchy are unequal because Anne Smith is specified as John Doe's manager.

> **Note:**
>
> In a query based on an implicit hierarchy, the client can specify in the search request the control 2.16.840.1.113894.1.8.3. The filter in this query specifies the attribute used to build the implicit hierarchy. For example, `(manager=cn=john doe, o=foo)` specifies the query for all people reporting directly or indirectly to John Doe. The implicit hierarchy is based on the `manager` attribute. The base of the search is ignored for such queries.
>
> For more information on controls used by Oracle Internet Directory, see About LDAP Controls in *Reference for Oracle Identity Management*.

> **See Also:**
>
> Overview of Oracle Internet Directory C API in *Application Developer's Guide for Oracle Identity Management*.

14.1.4 About Querying Group Entries

An application query list the members of the group and list of all groups.

An application can query either kind of group to do the following:

- List all members of a group
- List all groups of which a user is a member
- Check to see if a user is a member of a particular group

In addition, you can query dynamic groups, but not static ones, for whatever member attributes you specify.

> **Note:**
>
> The GSL_REQDATTR_CONTROL entry under LDAP Controls in *Reference for Oracle Identity Management*.

14.1.5 Understanding the `orclMemberOf` Attribute

`orclMemberOf` is a multivalued attribute containing the groups to which the entry belongs.

The following topics provide a conceptual description of the `orclMemberOf` attribute and also describe how to use this attribute in search filters:

- About `orclMemberOf` Attribute
- Examples of Using the `orclMemberOf` Attribute

14.1.5.1 About `orclMemberOf` Attribute

`orclMemberOf` is a multivalued attribute containing the groups to which the entry belongs. The groups in `orclMemberOf` include static groups and `labeleduri`-based dynamic groups. `CONNECT BY` assertion-based dynamic groups and dynamic lists are not included. The membership includes both direct groups and nested groups.

For example, suppose Mary is a member of the static group `directors` and the group `directors` is a member of the static group `managers`. If you do a specific query for the attribute `orclMemberOf` on Mary's DN, the values will contain both `managers` and `directors`.

The attribute values are computed during search and are not stored. `orclMemberOf` is not returned in a search unless explicitly requested by name.

As of Oracle Internet Directory 11g Release 1 (11.1.1.7.0), `orclMemberOf` can be used in search filters. Using `orclmemberof` is very useful with complex filters. Previously, the best way to find users belonging to multiple groups was to perform multiple search queries to fetch the data and then use client side application logic to compute the results. Now that `orclmemberof` can be used in search filters, you can do this with a single search query. Some of the examples include such queries.

`orclMemberOf` has the aliases `memberof` and `ismemberof` for compatibility with Active Directory and Oracle Directory Server Enterprise Edition (formerly Sun Java System Directory Server and SunONE iPlanet). These aliases can be used interchangeably and all of the search queries can be also be done using these two aliases instead of `orclmemberof`.

> **Note:**
>
> The attribute `orclMemberOf` is a virtual attribute, so it *cannot* be used for the following purposes:
>
> - In the LDAP filter part of the `labeleduri` attribute for creating dynamic groups or dynamic lists.
> - In a filter to be processed in memory using `orclinmemfiltprocess`. For information about `orclinmemfiltprocess`, see Optimizing Performance of Complex Search Filters in *Tuning Performance*.

14.1.5.2 Examples of Using the `orclMemberOf` Attribute

The following examples show how to use the `orclMemberOf` attribute in various search scenarios:

- Search single user:

```
ldapsearch -h host -p 3060 -D binddn -q -b "cn=jdoe,cn=users,o=oracle" -s base
"(objectclass=*)" orclmemberof
```

- Search with `memberof` alias:

```
ldapsearch -h host -p 3060 -D binddn -q -b "cn=jdoe,cn=users,o=oracle" -s base
"(objectclass=*)" memberof
```

- Get all attributes and `orclmemberof`:

```
ldapsearch -h host -p 3060 -D binddn -q -b "cn=jdoe,cn=users,o=oracle" -s base
"(objectclass=*)" orclmemberof  *
```

- Search multiple users:

```
ldapsearch -h host -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub
"(objectclass=person)" orclmemberof
```

- Use in search filters:
  - To determine whether user John Doe is member of the HR employees group, perform a base search against that user with a filter that checks if `orclmemberof` has the `HR` employees group.

    ```
    ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=johndoe,cn=users,o=oracle" -
    s base "(orclmemberof=cn=hr,cn=groups,o=oracle)" dn
    ```

  - To find all the users who are member of the `HR` employees group, perform a subtree search against `user` container for all entries with `orclmemberof` containing the `HR` employees group.

    ```
    ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub
    "(orclmemberof=cn=hr,cn=groups,o=oracle)" dn
    ```

  - To find the users who are member of multiple groups, use a filter that combines multiple `orclmemberof` conditions with `AND`. This provides the intersection of the two group memberships. The following search finds users who are members of both `HR` employees and `Managers` group.

    ```
    ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub
    "(&(orclmemberof=cn=hr,cn=groups,o=oracle)
    (orclmemberof=cn=managers,cn=groups,o=oracle))" dn
    ```

  - To find the users who are member of either `HR` employees or `Managers` group, use a filter that combines multiple `orclmemberof` conditions with `OR`. This provides the union of the two group memberships. You can use this method for three or more groups as well.

    ```
    ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub "(|
    (orclmemberof=cn=hr,cn=groups,o=oracle)
    (orclmemberof=cn=managers,cn=groups,o=oracle))" dn
    ```

  - To determine all employees who belong to `HR` group, have the location California, and have the title `HR Administrator`, use the following query:

    ```
    ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub
    "(&(orclmemberof=cn=hr,cn=groups,o=oracle)(l=ca)(title=HR Administrator))" dn
    ```

– The following example uses the `memberof` alias in a search filter instead of `orclmemberof`:

```
ldapsearch -h localhost -p 3060 -D binddn -q -b "cn=users,o=oracle" -s sub
"(memberof=cn=hr,cn=groups,o=oracle)" dn
```

> **Note:**
>
> Managing Alias Entries.

14.1.6 Considerations for Using Static and Dynamic Group

When deliberating about which kind of group to use, you must weigh the ease of administration against higher performance. For example, dynamic groups provide for easier administration, but cause a decrease in performance.

Table 14-2 lists some things to consider when deliberating whether to use static or dynamic groups.

Table 14-2 Static and Dynamic Group Considerations

Consideration	Static Groups	Dynamic Groups
Ease of administration	More difficult to administer if group memberships are large and change frequently	Easier to use, especially when group memberships are large and change frequently
Search Performance	Higher level of performance because you explicitly administer the membership list	Slightly decreased level of performance with dynamic groups using `labeleduri`, but almost same when compared to static groups, because memberships are cached. Decrease in performance with uncached groups, when compared to static groups and cached dynamic groups because memberships are computed as needed.

14.2 Managing Group Entries by Using Oracle Directory Services Manager

You can manage static and dynamic group entries by using the Data Browser page in Oracle Directory Services Manager.

You can display group entries, search for groups, and view groups using the procedures described in Managing Entries by Using Oracle Directory Services Manager. The procedures for creating and modifying groups are described in this section. This section contains the following topics:

- Creating Static Group Entries by Using Oracle Directory Services Manager
- Adding an Owner or Member to a Static Group Entry
- Modifying an Attribute of a Static Group Entry

- Creating Dynamic Group Entries by Using Oracle Directory Services Manager
- Adding an Owner or Member to a Dynamic Group Entry
- Modifying an Attribute of a Dynamic Group Entry
- Modifying a Dynamic Group Entry by Using Oracle Directory Services Manager

14.2.1 Creating Static Group Entries by Using Oracle Directory Services Manager

If the static group entry belongs to the `groupOfNames` object class, then you determine membership in the group by adding DNs to the multivalued attribute `member`. If the entry belongs to the `groupOfUniqueNames` object class, then you determine membership in the group by adding DNs to the multivalued attribute `uniqueMember`.

To add a static group entry:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. On the toolbar, choose the **Create a new entry** icon. Alternatively, right click any entry and choose **Create**.

   You can, alternatively, select a group that is similar to the one you want to create, then choose the **Create a new entry like this one** icon. Alternatively, right click any entry and choose **Create**.

   The Create New Entry wizard appears.
4. Specify the object classes for the new entry. Click the **Add** icon and use the Add Object Class dialog to select either `groupOfNames` or `groupOfUniqueNames`. (All the superclasses from this object class through `top` are also added.)

   Click **OK**.
5. In the **Parent of the entry** field, you can specify the full DN of the parent entry of the entry you are creating. You can also click **Browse** to locate and select the DN of the parent for the entry you want to add, then click **Select**.

   If you leave the **Parent of the entry** field blank, the entry is created under the root entry.
6. Click **Next**.
7. Choose an attribute which will be the **Relative Distinguished Name** value for this entry and enter a value for that attribute. You must enter a value for the `cn` attribute, even if it is not the RDN value.
8. Click **Next**. The next page of the wizard appears. (Alternatively, you can click **Back** to return to the previous page.)
9. Click **Finish**.

14.2.2 Adding an Owner or Member to a Static Group Entry

You can add an owner to or member to a static group entry.

Perform the following steps to add an owner or member to a static group entry:

1. Navigate to the group entry you just created in Creating Static Group Entries by Using Oracle Directory Services Manager.

2. Select the **Group** tab.
3. To add a member to the group:
   a. Click the Add icon next to the Members text box.
   b. Select the entry you want to add as a member (usually a user or group entry) in the Select Distinguished Name Path dialog.
   c. Click **OK**.
4. To add an owner to the group:
   a. Click the Add icon next to the Owners text box.
   b. Select the entry you want to add as an owner (usually a user or group entry) in the Select Distinguished Name Path dialog.
   c. Click **OK**.
5. Click **Apply** to save your changes or **Revert** to discard them.

14.2.3 Modifying an Attribute of a Static Group Entry

You can modify an attribute of a static group entry such as member list.

Perform the steps to modify an attribute, such as the member list, for a group entry:

1. Navigate to the group entry you just created in Creating Static Group Entries by Using Oracle Directory Services Manager.
2. Select the **Attributes** tab.
3. By default, only non-empty attributes are shown. You can switch between **Managed Attributes** and **Show All** by using the **Views** list.
4. To change the list of attributes shown as managed attributes:
   a. Click the icon under **Optional Attributes**.
   b. Select attributes you want to move from the All Attributes list to the Shown Attributes lists and use the **Move** and **Move All** arrows to move the attributes.

      Select attributes you want to move from the shown Attributes list to the All Attributes lists and use the **Remove** and **Remove All** arrows to move the attributes.
   c. Click **Add Attributes** to make your changes take effect or click **Cancel** to discard your changes. After you click **Add Attributes**, only the attributes that were on the Shown Attributes list are shown in the Managed Attributes view.
5. Specify values for the optional properties. You can also modify the values of the mandatory properties. For multivalued attributes, you can use the **Add** and **Delete** icons to add and delete multiple values.
6. Click **Apply** to save your changes or **Revert** to discard them.

   You can set an access control point (ACP) on this entry by using the Subtree Access and Local Access tabs. The procedures are described in Adding or Modifying an ACP by Using the Data Browser in ODSM and Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM.

See Also:

- Defining Dynamic Groups
- About Security Groups
- Globalization Support

14.2.4 Creating Dynamic Group Entries by Using Oracle Directory Services Manager

Dynamic groups can have static and dynamic members. The static members are listed as values of the `member` or `uniquemember` attribute. If the dynamic group entry belongs to the `groupOfNames` object class, then add static members to the group by adding DNs to the multivalued attribute `member`.

If the dynamic group entry belongs to the `groupOfUniqueNames` object class, then add static members to the group by adding DNs to the multivalued attribute `uniqueMember`.

For dynamic groups, you must also set attributes to specify how the group membership is computed. You must choose either the `labeledURI` or the `CONNECT BY` method for dynamically computing membership in the group. You cannot use both methods. If you are using the `labeledURI` method, you must set the `labeledURI` attribute, but not the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes. If you are using the `CONNECT BY` method, you must set the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes, but not the `labeledURI` attribute.

To add a dynamic group entry:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. On the toolbar, choose **Create a new entry**. The Create New Entry wizard appears.
4. Specify the object classes for the new entry. Select at least the following object class entries.
   - Either `groupOfNames` or `groupOfUniqueNames`
   - `orclDynamicGroup` or `orclDynamicList`

   Use `orclDynamicGroup` for a cached dynamic group based on `labeledURI` or a dynamic group based on `CONNECT BY`. Use `orclDynamicList` for an uncached dynamic list based on `labeledURI`.

   Click the **Add** icon and use the Add Object Class dialog to select object class entries. Optionally, use the search box to filter the list of object classes. To add the object class, select it and then click **OK**. (All the superclasses from this object class through `top` are also added.)
5. In the **Parent of the entry** field, you can specify the full DN of the parent entry of the entry you are creating. You can also click **Browse** to locate the DN of the parent for the entry you want to add, then click **Select**.

   If you leave the **Parent of the entry** field blank, the entry is created under the root entry.

6. Click **Next**.
7. Choose an attribute which will be the **Relative Distinguished Name** value for this entry and enter a value for that attribute. You must enter a value for the `cn` attribute, even if it is not the RDN value.
8. Click **Next**. The next page of the wizard appears. (Alternatively, you can click **Back** to return to the previous page.)
9. Click **Finish**.

14.2.5 Adding an Owner or Member to a Dynamic Group Entry

You can add an owner or member to a dynamic group entry.

To add an owner or member to a dynamic group entry:

1. Navigate to the group entry you just created in Creating Dynamic Group Entries by Using Oracle Directory Services Manager. (You might have to click the Refresh icon to see the new entry).
2. Select the **Group** tab.
3. To add an owner to the group:
   a. Click the **Add** icon next to the Owner box.
   b. Select the entry you want to add as owner (usually a user or group entry) in the Select Distinguished Name Path dialog.
   c. Click **OK**.
4. To add a member to the group:
   a. Click the **Add** icon next to the Members text box.
   b. Select the entry you want to add as a member (usually a user or group entry) in the Select Distinguished Name Path dialog.
   c. Click **OK**.
5. Choose **Apply** to apply your changes or choose **Revert** to abandon your changes.

14.2.6 Modifying an Attribute of a Dynamic Group Entry

You can modify an attribute for a dynamic group entry by using either `CONNECT BY` or `labeledURI` method for dynamically computing membership in the group.

To modify an attribute for a dynamic group entry:

1. Navigate to the group entry you just created in Creating Dynamic Group Entries by Using Oracle Directory Services Manager. (You might have to click the Refresh icon to see the new entry).
2. Select the **Attributes** tab.
3. You can switch between Managed Attributes and Show All by using the **Views** list.
4. To change the list of attributes shown as managed attributes:
   a. Click the icon under **Optional Attributes**.
   b. Select attributes you want to move from the All Attributes list to the Shown Attributes lists and use the **Move** and **Move All** arrows to move the attributes.

Select attributes you want to move from the shown Attributes list to the All Attributes lists and use the **Remove** and **Remove All** arrows to move the attributes.

c. Click **Add Attributes** to make your changes take effect or click **Cancel** to discard your changes. After you click **Add Attributes**, only the attributes that were on the Shown Attributes list are shown in the Managed Attributes view.

5. If you are using the `labeledURI` method for dynamically computing membership in the group, you must set the `labeledURI` attribute, but not the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes. In the **Attributes** tab page, in the `labeledURI` field, specify the following:

   `ldap:`*`ldap_URL`*

   For example:

   ```
   ldap://my_host:3000/ou=MyNeworganizationalUnit,
    o=MyCompany,c=US??sub?(objectclass=person)
   ```

   If you are using the `CONNECT BY` method for dynamically computing membership in the group, you must set the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes, but not the `labeledURI` attribute. In the `orclConnectByAttribute` field, specify the attribute that you want to use as the filter for the query—for example, `manager`. In the `orclConnectByStartingValue` field, specify the DN of the attribute you specified in the `orclConnectByAttribute` attribute—for example, `cn=Anne Smith`.

   For information about specifying the other attributes that appear in the **Attributes** tab page, see User and Group Schema Elements in *Reference for Oracle Identity Management*.

6. Click **Apply** to save your changes or **Revert** to discard them.

14.2.7 Modifying a Dynamic Group Entry by Using Oracle Directory Services Manager

Remember that you must choose either the `labeledURI` or the `CONNECT BY` method for dynamically computing membership in the group. You cannot use both methods. If you are using the `labeledURI` method, you must set the `labeledURI` attribute, but not the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes. If you are using the `CONNECT BY` method, you must set the `orclConnectByAttribute` and `orclConnectByStartingValue` attributes, but not the `labeledURI` attribute.

You can add static members to a dynamic group, but you are not required to do so.

You can set an access control point (ACP) on this entry by using the Subtree Access and Local Access tabs. The procedures are described in Adding or Modifying an ACP by Using the Data Browser in ODSM and Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM.

> **See Also:**
>
> - Defining Dynamic Groups
> - About Security Groups
> - Globalization Support

14.3 Managing Group Entries by Using the Command Line

You can manage static and dynamic groups from the command line by using LDAP tools.

This section contains the following topics:

- Creating a Static Group Entry by Using ldapadd
- Modifying a Static Group by Using ldapmodify
- Creating a Dynamic Group Entry by Using ldapadd
- Modifying a Dynamic Group by Using ldapmodify

> **Note:**
>
> - When you create a group, specifying members is optional and is shown here for the sake of completeness.
> - It is uncommon to have dynamic groups with static membership.

14.3.1 Creating a Static Group Entry by Using ldapadd

`ldapadd` is used to create a static group entry.

The syntax for the LDIF file is:

```
dn: DN_of_group_entry
objectclass: top
objectclass: groupOfNames | groupOfUniqueNames
member: DN of member 1
member: DN of member 2
.
.
.
member: DN of member N
```

The following command adds the group and members in this LDIF file to the directory:

```
ldapadd -p port_number -h host -D cn=orcladmin -q -f file_name.ldif
```

The following example shows an LDIF file named `myStaticGroup.ldif` for the entry for a group named MyStaticGroup:

```
dn: cn=myStaticGroup,c=us
objectclass: top
objectclass: groupOfNames
member: cn=John Doe
member: cn=Anne Smith
```

The following command adds the group and members in this LDIF file to the directory:

```
ldapadd -p 3060 -h myhost -D cn=orcladmin -q -f myStaticGroup.ldif
```

14.3.2 Modifying a Static Group by Using ldapmodify

`ldapmodify` is used to modify a static group.

To add a member to a group, the syntax of the LDIF file is:

```
dn: DN_of_group_entry
changetype: modify
add: member
member: DN of member entry
```

To delete a member from a group, the syntax of the LDIF file is:

```
dn: DN of group entry
changetype: modify
delete:member
member:DN of member entry
```

Issue this command to modify the file:

```
ldapmodify -D "cn=orcladmin" -q -p 3060 -v -f file_name.ldif
```

where -v specifies verbose mode.

The following example adds John Doe to a group named MyStaticGroup. As in the previous example, the data for this user entry is in the `myStaticGroup.ldif` file. This file contains the following:

```
dn: cn=myStaticGroup,c=us
changetype: modify
add:member
member: cn=John Doe
```

Issue this command to modify the file:

```
ldapmodify -D "cn=orcladmin" -q -p 3060 -v -f myStaticGroup.ldif
```

where -v specifies verbose mode.

> **Note:**
>
> When you add or modify an entry, the Oracle directory server does not verify the existence of the entry. However, if the attribute value must contain a DN, then the directory server verifies that the DN is specified.

14.3.3 Creating a Dynamic Group Entry by Using ldapadd

You can use `ldapadd` to create a dynamic group from the command line.

The following topics describe how to create a dynamic group using labeledURI attribute and CONNECT BY String:

- Creating a Cached Dynamic Group Using labeledURI Attribute
- Creating an Uncached Dynamic List Using labeledURI Attribute
- Creating a Dynamic Group Using CONNECT BY String

14.3.3.1 Creating a Cached Dynamic Group Using labeledURI Attribute

If you use the `labeledURI` attribute to create a cached dynamic group, then the syntax for the LDIF file is:

```
dn: DN_of_group_entry
objectclass: top
objectclass: groupOfNames | groupOfUniqueNames
objectclass: orcldynamicgroup
labeledURI:ldap:ldap_URL
member: DN of member 1
member: DN of member 2
.
.
.
member: DN of member N
```

Use the following command to add the group and members in this LDIF file to the directory:

```
ldapadd -p port_number -h host -f file_name.ldif
```

14.3.3.2 Creating an Uncached Dynamic List Using labeledURI Attribute

If you use the `labeledURI` attribute to create an uncached dynamic list, then the syntax for the LDIF file is:

```
dn: DN_of_group_entry
objectclass: top
objectclass: groupOfNames | groupOfUniqueNames
objectclass: orcldynamiclist
labeledURI:ldap:ldap_URL
member: DN of member 1
member: DN of member 2
.
.
.
member: DN of member N
```

Use the same command as in the previous example to add the group and members in this LDIF file to the directory:

```
ldapadd -p port_number -h host -f file_name.ldif
```

14.3.3.3 Creating a Dynamic Group Using CONNECT BY String

If you use the `CONNECT BY` string, then the syntax for the LDIF file is:

```
dn: DN_of_group_entry
objectclass: top
objectclass: groupOfNames | groupOfUniqueNames
objectclass: orclDynamicGroup
orclConnectByAttribute:attribute_name
orclConnectByStartingValue:DN_of_attribute
member: DN of member 1
member: DN of member 2
.
.
.
member: DN of member N
```

When specifying entries in this syntax, do not use double quotes around distinguished names.

The following example shows an LDIF file for the entry for a dynamic group:

```
dn: cn=myDynamicGroup,c=us
objectclass: top
objectclass: groupOfNames
objectclass: orcldynamicgroup
labeledURI:ldap://my_host:3000/ou=MyNeworganizationalUnit,
 o=MyCompany,c=US??sub?(objectclass=person)
member: cn=John Doe
member: cn=Anne Smith
```

The following command adds this LDIF file to the directory:

```
ldapadd -p 3060 -h myhost -f myDynamicGroup.ldif
```

14.3.4 Modifying a Dynamic Group by Using ldapmodify

`ldapmodify` command is used to modify a dynamic group.

To change the organizational unit of the group created in the previous example, the syntax of the LDIF file is:

```
dn: DN_of_group_entry
changetype: modify
replace:labeledURI
labeledURI:ldap://my_host:3000/
 ou=MyNeworganizationalUnit,o=MyCompany,c=US??sub?(objectclass=person)
```

> **Note:**
>
> When you add or modify an entry, the Oracle directory server does not verify the syntax of the attribute values in the entry.

15

Performing Bulk Operations

You can perform bulk operations for Oracle Internet Directory to manage object classes, attributes, attribute aliases, and attribute indexes. You can change the server mode and use tools such as bulkload, bulkmodify, bulkdelete, the Catalog management tool, and LDAP command-line utilities to perform bulk operations.
This chapter includes the following sections:

- Introduction to Performing Bulk Operations
- Setting Environment Variables Before Using Command-line Tools
- Changing the Server Mode
- Loading Data Into the Schema by Using bulkload
- Modifying Attributes By Using bulkmodify
- Deleting Entries by Using bulkdelete
- Dumping Data from Oracle Internet Directory to a File by Using ldifwrite
- Creating and Dropping Indexes from Existing Attributes by Using catalog

15.1 Introduction to Performing Bulk Operations

For processing large quantities of data, bulk operations are typically more efficient than standard LDAP operations You can perform bulk operations only by using the command-line bulk tools.

> **Note:**
>
> - The bulk tools do not support attribute uniqueness.
> - If your schemas were created during installation of a version prior to 11g Release 1 (11.1.1.6.0), you must add datafiles to the OLTS_CT_STORE and OLTS_ATTRSTORE tablespaces if you intend to add more than a million entries to Oracle Internet Directory. Perform this step prior to the `bulkload` or `ldapadd` operation. For details, see *Oracle Database Administrator's Guide*.

In addition to this chapter, bulk tools are also discussed in the following sections of this book:

- Indexing an Attribute by Using the Catalog Management Tool
- Migrating Data from LDAP-Compliant Directories
- Setting Up Replication

See Also:

The chapter on Oracle Internet Directory in *Reference for Oracle Identity Management*.

Note:

- Stop all instances of Oracle Internet Directory before using `bulkload`. Before using any of the other bulk tools, either stop all instances of Oracle Internet Directory or disable the entry cache.
- Do not start Oracle Internet Directory while a bulk tool is executing.

Tip:

The syntax descriptions in this chapter show true or false arguments in the following format:

```
check="TRUE"
```

You can type `true` or `false` in uppercase or lowercase, you can specify just the first letter, and you can omit the double quotes. That is, the following specifications are equivalent:

```
check=t
check=true
check="true"
check=T
check=TRUE
check="TRUE"
```

15.2 Setting Environment Variables Before Using Command-line Tools

Before you begin using the Oracle Identity Management command-line tools, you must configure your environment. This involves setting the appropriate environment variables.

The syntax and examples provided in this guide require that you have the following environment variables set:

- `ORACLE_HOME` - The location of non-writable files in your Oracle Identity Management installation.
- `DOMAIN_HOME` - The location of writable files in your Oracle Identity Management installation.
- `NLS_LANG` (*`APPROPRIATE_LANGUAGE`*`.AL32UTF8`) - The default language set at installation is `AMERICAN_AMERICA`.

- `WLS_HOME` - The location where the WebLogic Server is installed. This environment variable is required for Oracle Directory Integration Platform commands but not Oracle Internet Directory commands.
- `PATH` - The following directory locations should be added to your `PATH`:

  $ORACLE_HOME`/bin`

  $ORACLE_HOME`/ldap/bin`

  $ORACLE_HOME`/ldap/admin`

15.3 Changing the Server Mode

Some bulk tool operations and replication setup procedures require you to switch an Oracle Internet Directory instance from read/write to read-only mode or from read-only to read/write mode. You can do so by using either Oracle Enterprise Manager Fusion Middleware Control or `ldapmodify`.

While the server is in read-only mode, only the administrator `cn=orcladmin` can write to the directory. As a result, you can only make changes to the directory by using utilities that enable you to connect as `cn=orcladmin`. These include:

- Oracle Directory Services Manager
- Command line tools

You cannot make changes while the server is in read-only mode by using WLST or Oracle Enterprise Manager Fusion Middleware Control because they connect to the Oracle Internet Directory server as the user `cn=emd admin, cn=oracle internet directory`.

The following sections explain this further:

- Setting the Server Mode by Using Fusion Middleware Control
- Setting the Server Mode by Using ldapmodify

15.3.1 Setting the Server Mode by Using Fusion Middleware Control

You can set the server mode to read-only from Fusion Middleware Control.

To set the server mode to read-only from perform the following steps:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu.
2. Select **General**.
3. Choose **Read-only** for Server Mode.
4. Click **Apply**.

To set the server mode to read/write mode from Fusion Middleware Control, use the same procedure but choose `Read/Write` in Step 3.

15.3.2 Setting the Server Mode by Using ldapmodify

You can set the server mode to Read-Only by using `ldapmodify`.

Run the following commands to set the server mode:

```
ldapmodify -D "cn=orcladmin" -q -h host_name \
           -p port -f change_mode.ldif
```

where the file change_mode.ldif has the following content:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclservermode
orclservermode: r
```

To set the server mode to Read/Write by using `ldapmodify`, use the same command but change the last line in the LDIF file to:

```
orclservermode: rw
```

15.4 Loading Data Into the Schema by Using bulkload

The bulk loader, `bulkload`, is a bulk management tool. It takes input data in LDIF or SQL*Loader format and loads the data directly into Oracle Internet Directory's schema in the metadata repository.

> **Note:**
>
> - If a directory server instance is participating in a replication agreement, do not use the `bulkload` tool to add data into the node. Instead, use `ldapadd`.
> - Before using `bulkload`, ensure that the environment variable `DOMAIN_HOME` has been set to the full path name of the Oracle instance.
> - Running the `bulkload load` operation sets the server mode to read/write. If you require a different mode, reset it after performing the `load` operation.
> - If the applicable password policy has the `pwdmustchange` attribute set to 1, then for every new entry loaded by `bulkload`, the `pwdreset` attribute is set to 1 by default. See Managing Password Policies for more information.
> - If you do not use the `bulkload` utility to populate the directory, then you must run the `oidstats.sql` tool to avoid significant search performance degradation.

> **See Also:**
>
> The `oidstats.sql` command-line tool reference in *Reference for Oracle Identity Management* for a description and syntax for the oidstats.sql tool

The following sections explain this further:

- Different Phases of Loading Data
- Output File Locations for bulkload Tool
- Running the bulkload Tool
- Importing an LDIF File by Using bulkload
- Loading Data in Incremental or Append Mode By Using bulkload
- Performing Index Verification By Using bulkload
- Re-Creating Indexes By Using bulkload

- Recovering Data After a Load Failure By Using bulkload

15.4.1 Different Phases of Loading Data

You can load the data in `check`, `generate` and `load` phases.

In the `check` phase, `bulkload` parses and verifies the LDIF input data for the schema.

In the `generate` phase, `bulkload` generates intermediate files in SQL*Loader format.

In the `load` phase, `bulkload` can use either of two methods: bulk mode loading or incremental mode loading.

- When using bulk mode loading, `bulkload` loads generated intermediate files into the database. As it does so, it drops old indexes and generates new ones.
- When using incremental mode loading, `bulkload` loads intermediate files to tables in the database using insert mode. While it loads the data, bulkload updates the indexes.

Bulk mode loading is faster than incremental mode loading.

The bulk loader also supports the following features:

- It enables you to specify the number of threads in order to achieve parallelism during the generate and load phases.
- It has an `encode` option that enables you to use data in other languages.
- It has a `restore` option that enables you to retain the operational attributes specified in the LDIF file.
- It has an `index` option for index recreation and a `missing` option for creation of missing indexes.
- It has a `recover` option that is useful for recovering from `bulkload` failures.
- When appending the data to existing directories, `bulkload` supports both bulk mode and incremental mode loading.
- The `append` option enables you to load data while the LDAP server is up and running.

At the beginning of the `generate` phase, the server's `orclServerMode`, in the instance-specific configuration entry, is changed from `read/write` to `read-modify`. At the end of the `generate` phase, it is left in the `read-modify` state so that you cannot add entries to Oracle Internet Directory between the `generate` and `load` phases. This is necessary to maintain internal sequence numbers. You are expeced to run the `load` phase immediately after the `generate` phase. At the end of the `load` phase, the servers' `orclServerMode` is set back to `read/write`. Using `bulkload` with the `recover` option also sets `orclServerMode` back to `read/write`.

At the start of the load operation, `bulkload` determines the current configured value of `orclRIenabled`, then disables referential integrity. At the end of load phase, bulkload returns `orclRIenabled` to its original value. If is any referential integrity violations occurred, however, referential integrity is disabled, and you see the message:

```
There is a violation of Referential Integrity and hence it is Disabled now. Run the
OIDDIAG tool with diagnostic option to collect the Entries which have dangling DN
attribute values and Fix the violation
```

Fix the violation and then set orclRIenabled to the desired value.

15.4.2 Output File Locations for bulkload Tool

The `bulkload` tool generates output log, a list of duplicate DNs and intermediate files.

The `bulkload` tool generates the following output files in the *`$DOMAIN_HOME`*`/tools/OID/logs` directory:

- An output log, `bulkload.log`
- A list of duplicate DNs, `duplicateDN.log`
- Intermediate files, `*.ctl` and `*.dat`

The `bulkload` tool generates the following output files in the *`$DOMAIN_HOME`*`/tools/OID/load` directory:

- A list of bad LDIF entries, `badentry.ldif`
- A list of all dynamic group entries that can be added using `ldapadd`, `dynGrp.ldif`
- Intermediate log files generated by the SQL*Loader, `bsl_*.log`

15.4.3 Running the bulkload Tool

You must set the environment variable `DOMAIN_HOME` while you are running `bulkload` Tool.

The `bulkload` tool has the following syntax:

```
bulkload [connect=connect_string]
{[check="TRUE"|"FALSE" [file=ldif_file]] [generate="TRUE"|"FALSE"
[append="TRUE"|"FALSE"] [restore="TRUE"|"FALSE"] [thread=num_of_threads]
file=ldif_file] [load="TRUE"|"FALSE" [append="TRUE"|"FALSE"]
[threads=num_of_threads]] [index="TRUE"|"FALSE"] [missing="TRUE"|"FALSE"]
[recover="TRUE"|"FALSE"]} [encode=character_set] [debug="TRUE"|"FALSE"]
[verbose="TRUE"|"FALSE"]
```

Some of the parameter combinations are valid while others are invalid.

You must set the environment variable `DOMAIN_HOME`. Specify the fully qualified path to the Oracle Instance where the intermediate file is generated.

You must specify at least one of the following actions when you invoke `bulkload`: `check`, `generate`, `load`, `append`, `recover`, or `index`.

If `check` is `TRUE`, `bulkload` performs a schema check.

If `generate` is `TRUE`, `bulkload` generates intermediate files.

When using the `check` or `generate` action, you must specify the path name to the LDIF data file.

If `load` is `TRUE`, `bulkload` loads intermediate files.

When `append` is `TRUE`, `bulkload` can perform its actions while the server is up and running.

Use the `restore` flag only when the LDIF file contains operational attributes, such as `orclguid` or `creatorsname`. Avoid having operational attributes in the LDIF file when the `restore` flag is not specified or is set to FALSE.

Do not specify `recover` with any other option.

The option combination `check index` verifies the existing indexes.

15.4.4 Importing an LDIF File by Using bulkload

You can use `bulkload` utility to import an LDIF file.

This section discusses the tasks to process an LDIF file through `bulkload`.

> **See Also:**
>
> The `bulkload` command-line tool reference in *Reference for Oracle Identity Management*.

This section contains these topics:

- Stopping Oracle Internet Directory Processes
- Backing Up the Oracle Database Server
- Finding Out the Oracle Internet Directory Password
- Checking Input File and Generating Files for SQL*Loader
- Loading Input Files

15.4.4.1 Stopping Oracle Internet Directory Processes

Stop all Oracle Internet Directory server instances by using either Fusion Middleware Control or the command line tool `wlst` command as gvien below:

```
shutdown(name='instance-name',type='OID')
```

> **See Also:**
>
> Managing Oracle Internet Directory Instances

15.4.4.2 Backing Up the Oracle Database Server

Before you import the file, back up the Oracle database server as a safety precaution.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* in the Oracle Database Documentation Library

15.4.4.3 Finding Out the Oracle Internet Directory Password

To use `bulkload`, you must provide the Oracle Internet Directory `ODS` schema password.

> **See Also:**
>
> The `oidpasswd` command-line tool reference in *Reference for Oracle Identity Management* .

15.4.4.4 Checking Input File and Generating Files for SQL*Loader

On UNIX, the `bulkload` tool usually resides in `$ORACLE_HOME/ldap/bin`. On Microsoft Windows, this tool usually resides in `$ORACLE_HOME\ldap\bin`.

Check the input file and generate files for the SQL*Loader by typing:

```
bulkload connect="connect_string" \
   check="TRUE" generate="TRUE" file="full_path_to_ldif-file_name"
```

When you specify both the `check` and `generate` options, the entries are checked for schema compliance.

All `check`-related errors are reported as command line output. All schema violations are reported in *`$DOMAIN_HOME`*`/tools/OID/logs/bulkload.log`. All bad entries are logged in *`$DOMAIN_HOME`*`/OID/load/badentry.ldif`.

If there are duplicate entries, their DNs are logged in *`$DOMAIN_HOME`*`/tools/OID/logs/duplicateDN.log`. This is just for information purpose. The `bulkload` tool does not generate duplicate data for duplicate entries. It ignores duplicate entries.

Use a text editor to fix all bad entries, then re-run `bulkload` with the `check` and `generate` options. Repeat until there are no errors, or until the remaining errors are acceptable to you. For example, you might be willing to load a small number of entries with `ldapadd`.

The `bulkload` tool generates intermediate `*.ctl` and `*.dat` files in the *`$DOMAIN_HOME`*`/OID/load` directory. Even when errors occur, `bulkload` generates the intermediate files for those entries that had no `check` errors.

When bulkload completes successfully or with acceptable errors, you can use the intermediate files with the SQL*Loader in `load` mode. Do not modify these files.

> **Note:**
>
> Always use the `check` and `generate` options together if you plan to ignore `check`-related errors. If you use the `generate` option without the `check` option, none of the validation checks are performed. In that case, the intermediate files will contain erroneous entries. Loading such files can lead to data inconsistency and index creation failures.

15.4.4.5 Loading Input Files

After you have generated the input files, run `bulkload` with the `load` option. During this step, `bulkload` loads the `*.dat` files, which are in Oracle SQL*Loader specific format, into the database, creates attribute indexes, and generates database statistics. The syntax is:

```
bulkload connect="connect_string" load="TRUE"
```

The tool will indicate any errors on the screen. All loading errors are reported in the *$DOMAIN_HOME*/tools/OID/logs directory. They reside in bulkload.log and in the SQL*Loader-generated files *.bad and bsl_*.log.If load fails, the database might be in an inconsistent state. Restore the database to its state prior to the bulkload operation, either by using bulkload with the recover option or by restoring Oracle Internet Directory directory from a backup taken before you invoked bulkload. Then repeat the command:

```
bulkload connect="connect_string" load="TRUE"
```

If you encounter an error during the indexing phase, you can use:

```
bulkload coonect="con_str" index=true
```

to re-create all indexes.

If you encounter an error during database statistics generation, you can use the oidstats.sql command to generate statistics.

> **See Also:**
>
> The oidstats.sql command reference in *Reference for Oracle Identity Management*.

15.4.5 Loading Data in Incremental or Append Mode By Using bulkload

If you must add entries to an Oracle Internet Directory server that already contains data, and the server must be up and running at the same time, then you must use the incremental or append mode. This mode is usually faster than other methods of adding entries to the directory. However, you must ensure that the Oracle Internet Directory LDAP instances are in read-modify mode so that bulkload can append data.

You invoke bulkload in incremental or append mode with command lines similar to these:

```
bulkload connect="conn_str" \
   check="TRUE" generate="TRUE" append="TRUE" file="LDIF_file"
bulkload connect="conn_str" \
    load="TRUE" append="TRUE"
```

15.4.6 Performing Index Verification By Using bulkload

The bulkload operation can either update indexes or create indexes. Sometimes, however, bulkload does not update or create the indexes properly.

This is typically due to issues like improper sizing. If this happens, you can use bulkload to verify and re-create all the indexes.

Use the following syntax to invoke bulkload for verification of indexes:

```
bulkload connect="conn_str" \
    check="TRUE" index="TRUE"
```

15.4.7 Re-Creating Indexes By Using bulkload

You can re-create indexes using `bulkload` tool.

Use the following syntax to re-create index:

```
bulkload connect="conn_str" index="TRUE"
```

15.4.8 Recovering Data After a Load Failure By Using bulkload

The `load` phase of bulkload can fail because of issues like improper disk sizing. After such a failure, the directory data might be inconsistent.

You can use the `recover` option to return the directory data to its pre-bulkload state. The syntax is:

```
bulkload connect="conn_str" recover="TRUE"
```

15.5 Modifying Attributes By Using bulkmodify

The `bulkmodify` tool is useful for modifying the attributes of a large number of entries in an existing directory. It can perform add and replace operations on attribute values. It can operate on a naming context. Using filters, it can also operate selectively on a few entries under a specified naming context.

> **See Also:**
>
> The `bulkmodify` command-line tool reference in *Reference for Oracle Identity Management*

> **Note:**
>
> Before using `bulkmodify`, ensure that the environment variable `DOMAIN_HOME` has been set to the full path name of the Oracle instance.

The following sections explain this further:

- Attributes Excluded from bulkmodify Operations
- Log File Location for bulkmodify Tool
- Running the bulkmodify Tool
- Adding a Description for All Entries Under a Specified Naming Context
- Adding an Attribute for Specific Entries Under a Specified Naming Context
- Replacing an Attribute for All Entries Under a Specified Naming Context

15.5.1 Attributes Excluded from bulkmodify Operations

There are certain attributes which are excluded from `bulkmodify` operations.

The `bulkmodify` tool does not allow `add` or `replace` operations on the following attributes:

- `dn` (use `ldapmoddn` instead)
- `cn` (use `ldapmodify` instead)
- `userpassword` (use `ldapmodify` instead)
- `orclpassword` (use `ldapmodify` instead)
- `orclentrylevelaci` (use `ldapmodify` instead)
- `orclaci` (use `ldapmodify` instead)
- `orclcertificatehash`
- `orclcertificatematch`
- any binary attribute
- any operational attribute

It does not allow `replace` operation on the attribute `objectclass`.

It does not allow `add` for single-valued attributes.

15.5.2 Log File Location for bulkmodify Tool

Output from `bulkmodify` is logged in *`$DOMAIN_HOME`*`/tools/OID/logs/bulkmodify.log`.

15.5.3 Running the bulkmodify Tool

You can run `bulkmodify` tool with `add` or `replace` option.

`bulkmodify` has the following syntax.

```
bulkmodify connect=connect_string basedn=Base_DN
{[add="TRUE"|"FALSE"]|[replace="TRUE"|"FALSE"]} attribute=attribute_name
value=attribute_value [-q] [filter=filter_string] [size=transaction_size]
[threads=num_of_threads] [debug="TRUE"|"FALSE"] [encode=character_set]
[verbose="TRUE"|"FALSE"]
```

The number of threads should be from one to six times the number of processors.

Select either the `add` or the `replace` option. By default both are set to `FALSE`.

15.5.4 Adding a Description for All Entries Under a Specified Naming Context

You can add descriptions for entries under a specified naming context.

This example adds descriptions for all the entries under `"c=us"`.

```
bulkmodify connect="connect_str" basedn="c=us" add="TRUE" \
    attribute="description" value="US citizen" filter="objectclass=*"
```

15.5.5 Adding an Attribute for Specific Entries Under a Specified Naming Context

You can add an attribute for specific entries under a specified naming context.

This example adds `telephonenumber` for all the entries under `"c=us"` that have the manager Anne Smith.

```
bulkmodify connect="connect_str" basedn="c=us" add="TRUE" \
     attribute="telephoneNumber" \
     value="408-123-4567" filter="manager=cn=Anne Smith"
```

15.5.6 Replacing an Attribute for All Entries Under a Specified Naming Context

You can replace an attribute for all entries under a specified naming context by running `bulkmodify` command.

This example replaces `pwdreset` for all the entries under `"c=us"`.

```
bulkmodify connect="connect_str"  basedn="c=us" replace="TRUE" \
    attribute="pwdreset" value="1" filter="objectclass=*"
```

15.6 Deleting Entries by Using bulkdelete

bulkdelete is useful for deleting a large number of entries in an existing directory. bulkdelete can delete entries specified under a naming context. By default, it deletes the entries completely. It removes all traces of an entry from the database.

> **See Also:**
>
> The `bulkdelete` command-line tool reference in *Reference for Oracle Identity Management*.

> **Note:**
>
> - Before using `bulkdelete`, ensure that the environment variable `DOMAIN_HOME` has been set to the full path name of the Oracle instance.
> - If the number of entries to be deleted in the specified naming context is large, you must tune the Oracle Database as specified in the Oracle Internet Directory Performance Tuning in *Tuning Performance*.

The following sections explain this further:

- Log File Location for bulkdelete Tool
- Running the bulkdelete Tool

- Deleting All Entries Under a Specified Naming Context by Using bulkdelete
- Deleting Entries Under a Naming Context and Making them Tombstone Entries
- Deleting All Entries Under a Specified Subtree by Applying the Filter Parameter

15.6.1 Log File Location for bulkdelete Tool

Bulkdelete output is logged in *$DOMAIN_HOME*/tools/OID/logs/bulkdelete.log.

15.6.2 Running the bulkdelete Tool

The `bulkdelete` tool is used to remove the entries from database.

The `bulkdelete` tool has the following syntax.

```
bulkdelete connect=connect_string {[basedn=Base_DN] | [file=file_name]}
[filter="LDAP_search_filter"] [encode=character_set] [debug="TRUE"|"FALSE"]
[cleandb="TRUE"|"FALSE"] [skipcheck="TRUE"|"FALSE"] [size=transaction_size]
[threads=num_of_threads] [verbose="TRUE"|"FALSE"]
```

Select either the `basedn` or the `file` option. If `cleandb` is `TRUE`, `bulkdelete` removes entries completely from the database. By default `cleandb` is set to `TRUE`. If you use the option `cleandb FALSE`, `bulkdelete` turns all entries into tombstone entries instead of deleting them completely. The number of threads should be from one to six times the number of CPUs.

15.6.3 Deleting All Entries Under a Specified Naming Context by Using bulkdelete

You can delete all entries under a specified naming context by using `bulkdelete` command.

This example deletes all the entries under "`c=us`".

```
bulkdelete connect="connect_str" basedn="c=us" cleandb="TRUE"
```

15.6.4 Deleting Entries Under a Naming Context and Making them Tombstone Entries

You can delete the entries under a naming context and make them Tombstone entries.

This example deletes all the entries under "`c=us`" and leaves them as tombstone entries.

```
bulkdelete connect="connect_str" basedn="c=us" cleandb=FALSE
```

This example deletes all the entries under given base DNs specified in file and leaves them as tombstone entries.

```
bulkdelete connect="connect_str" file="file" cleandb=FALSE
```

15.6.5 Deleting All Entries Under a Specified Subtree by Applying the Filter Option

You can delete all entries under a specified Subtree by specifying the `filter` option in the `bulkdelete` command. This option enables you to delete a huge amount of data in a reasonable amount of time.

This example deletes all the "`c=us`" entries by matching the `modifyTimestamp`.

```
bulkdelete connect="connect_str" basedn="c=us"
filter="(&(cn=user*)(modifytimestamp>=20180123101234z))" cleandb="TRUE"
verbose="TRUE" threads=5 size=10000 skipcheck="TRUE"
```

15.7 Dumping Data from Oracle Internet Directory to a File by Using ldifwrite

The `ldifwrite` tool is used to dump of data from an Oracle Internet Directory store to a file. Having the data in a file facilitates loading the data into another node for replication or backup storage. As it writes to the output file, the `ldifwrite` tool performs a subtree search, including all entries below the specified DN, and the DN itself. It dumps data in LDIF format. It can also dump entries under a specified replication agreement DN.

The `ldifwrite` tool can dump entries located by using specified filters.

> **See Also:**
>
> The `ldifwrite` command-line tool reference in *Reference for Oracle Identity Management*.

> **Note:**
>
> Before using `ldifwrite`, ensure that the environment variable `DOMAIN_HOME` has been set to the full path name of the Oracle instance.

The following sections explain this further:

- Log File Location for ldifwrite Tool
- Running the ldifwrite Tool
- Dumping Part of a Specified Naming Context to an LDIF File
- Dumping Entries Under a Specified Naming Context to an LDIF File

15.7.1 Log File Location for ldifwrite Tool

Output from `ldifwrite` is logged in *`$DOMAIN_HOME`*`/tools/OID/logs/ldifwrite.log`.

15.7.2 Running the ldifwrite Tool

You can run `ldifwrite` in the backend.

The `ldifwrite` tool has the following syntax.

```
ldifwrite connect=connect_string basedn=Base_DN ldiffile=LDIF_Filename
[filter=LDAP_Filter] [threads=num_of_threads] [debug="TRUE"|"FALSE"]
[encode=character_set] [verbose="TRUE"|"FALSE"]
```

Use the `basedn` option to specify the base DN or replication agreement DN.

The number of threads should be from one to six times the number of CPUs.

15.7.3 Dumping Part of a Specified Naming Context to an LDIF File

The Naming Context objects can be dumped into an LDIF file by running commands.

This example uses the following naming context objects defined in partial replication:

```
dn: cn=includednamingcontext000001, cn=replication namecontext,
 orclagreementid=000001, orclreplicaid=node replica identifier,
 cn=replication configuration
orclincludednamingcontexts: c=us
orclexcludednamingcontexts: ou=Americas, c=us
orclexcludedattributes: userpassword
objectclass: top
objectclass: orclreplnamectxconfig
```

In this example, all entries under "`c=us`" are backed up except "`ou=Americas,c=us`". The `userpassword` attribute is also excluded. The command is:

```
ldifwrite connect="conn_str"  \
  basedn="cn=includednamingcontext000001, cn=replication namecontext, \
  orclagreementid=000001,orclreplicaid=node replica identifier,\
  cn=replication configuration" ldiffile="ldif_file_name"
```

15.7.4 Dumping Entries Under a Specified Naming Context to an LDIF File

You can write all the entries with specific LDAP search filter by running certain commands.

This example writes all the entries that satisfy LDAP search filter criteria under "`ou=Europe, o=imc, c=us`" into the `output.ldif` file.

```
ldifwrite connect="connect_str" basedn="ou=Europe, o=imc, c=us"\
   filter="uid=abc" ldiffile="output.ldif"
```

15.8 Creating and Dropping Indexes from Existing Attributes by Using catalog

You can use the autocatalog feature, which is enabled by default in fresh installs. You can also enable it if you have upgraded from a previous release. When this feature is enabled, Oracle Internet Directory automatically invokes the `catalog` command to index attributes when you search for them. If the autocatalog feature is not enabled, and you want to use previously

uncataloged attributes in search filters, you must add them to the catalog entry, as in previous releases.

You can now use `ldapmodify` to create and drop indexes. See Indexing an Attribute by Using ldapmodify. The `ldapmodify` command invokes `catalog` to perform the operation. You can still use catalog for this purpose.

The `catalog` tool is useful for creating indexes for or dropping indexes from existing attributes. The catalog tool makes an attribute searchable.

> **Note:**
>
> - You can use the LDAP tool `ldapmodify` to create and drop indexes from attributes. See Indexing an Attribute by Using ldapmodify. The `ldapmodify` tool actually invokes `catalog`, and you can still use `catalog` for this purpose.
> - Before using `catalog`, ensure that the environment variable `DOMAIN_HOME` has been set to the full path name of the Oracle instance.
> - The `catalog` command cannot index more than 1000 attributes at a time. If more than 1000 attributes are present in the file, the tool throws an error. If you need to index more than 1000 attributes, use multiple files.
> - You can specify the IOT option to improve performance. For more information, see "Oracle Internet Directory Data Management Tools" in *Reference for Oracle Identity Management*.

The following sections explain this further:

- Log File Location for catalog Tool
- Running the catalog Tool
- Changing a Searchable Attribute into a Non-searchable Attribute
- Changing a Non-searchable Attribute into a Searchable Attribute

15.8.1 Log File Location for catalog Tool

Output from `catalog` is logged in *`$DOMAIN_HOME`*`/tools/OID/logs/catalog.log`.

15.8.2 Running the catalog Tool

The catalog tool contains the values `add`, `attribute` and `logging`.

catalog has the following syntax.

```
catalog connect=connect_string {[add="TRUE"|"FALSE"]|[delete="TRUE"|"FALSE"]}
{[attribute=attribute_name]|[file=file_name]} [logging="TRUE"|"FALSE"]
[threads=num_of_threads] [debug="TRUE"|"FALSE"] [iot="TRUE"|"FALSE"]
[verbose="TRUE"|"FALSE"]
```

Select either the `add` or the `delete` option. By default both are set to `FALSE`.

The number of threads should be from one to six times the number of CPUs.

If `logging` is `TRUE`, `catalog` generates a redo log.

You can specify only one `attribute` argument on the command line at a time. To add or delete more than one attribute in a single command invocation, use the `file` option and specify a list of attributes in the file. Use one line for each attribute, for example:

```
description
sn
title
```

15.8.3 Changing a Searchable Attribute into a Non-searchable Attribute

A searchable attribute can be changed into a non-searchable attribute.

This example drops an index from the attribute `title`.

```
catalog connect="connect_str" delete="TRUE" attribute="title"
```

> **Note:**
>
> Unless you are absolutely sure that the indexes were not created by the base schema that was installed with Oracle Internet Directory, be careful not to use the `catalog delete=T` option to remove indexes from attributes. Removing indexes from base schema attributes can adversely impact the operation of Oracle Internet Directory.

15.8.4 Changing a Non-searchable Attribute into a Searchable Attribute

You can change a non-searchable attribute into a searchable attribute.

This example adds an index to the attribute `title`.

```
catalog connect="connect_str" add="TRUE" attribute="title"
```

> **See Also:**
>
> - Indexing an Attribute by Using the Catalog Management Tool
> - Adding an Index to a New Attribute by Using Oracle Directory Services Manager
> - Adding an Index to an Existing Attribute by Using Oracle Directory Services Manager
> - Configuring Specific Attributes for Referential Integrity by Using the Command Line

16

Managing Collective Attributes

The collective attributes in Oracle Internet Directory can be managed by using LDAP command-line tools.
This chapter includes the following sections:

- Introduction to Collective Attributes
- Managing Collective Attributes by Using the Command Line

16.1 Introduction to Collective Attributes

Attributes shared by the entries comprising an entry collection are called collective attributes. Values of collective attributes are visible but not updatable to clients accessing entries within the collection.

As administrator, you manage collective attributes by defining and modifying the associated collective attributes sub entry.

> **See Also :**
>
> RFC 3671 "Collective Attributes in the Lightweight Directory Access Protocol (LDAP)" and RFC 3672 "Subentries in the Lightweight Directory Access Protocol (LDAP)" at `http://www.ietf.org` for more information.

- RFC Definition and Oracle Extensions
- Defining the Collective Attribute Subentry
- Using subtreeSpecification Attribute
- Overriding a Collective Attribute

16.1.1 RFC Definition and Oracle Extensions

RFC 3671 describes a specific schema for collective attributes.

If you want to, you can define and use the collective attribute schema exactly as described in the RFC. Oracle Internet Directory, however, extends the definition of collective attributes to make them easier to use.

16.1.1.1 RFC 3671

According to RFC 3671, you must define each collective attribute schema before you can use it in the collective subentry. For example, if you want to use the telephone number attribute as a collective attribute, then you define the schema for the `c-telephoneNumber` in the directory like this:

```
( 2.5.4.20.1 NAME 'c-TelephoneNumber' SUP telephoneNumber COLLECTIVE )
```

In addition, the collective attribute must be multivalued.

16.1.1.2 Oracle Extensions to RFC 3671

Oracle extends the usage as follows:

- You do not need a schema definition for a collective attribute. You can use any attribute as a collective attribute.
- To make an attribute a collective attribute, create it in the collective subentry with the subtype `collective`.
- A collective attribute can be either multivalued or single valued.

The rest of this chapter describes Oracle Internet Directory usage.

16.1.2 Defining the Collective Attribute Subentry

You create a collective attribute by defining a subentry. The following example defines a collective subentry for the entries under `dc=mycompany,dc=com`.

This subentry causes `TelephoneNumber` and `postalCode` to be included as collective attributes in all the entries under `dc=mycompany,dc=com`:

```
Dn: cn=collective attributes, dc=mycompany,dc=com
Cn: collective attributes
Objectclass: subentry
Objectclass: collectiveAttributeSubentry
Objectclass: top
Objectclass: extensibleobject
TelephoneNumber;collective: 1234560000
PostalCode;collective: 98765
```

16.1.3 Using subtreeSpecification Attribute

You can control which specific entries actually get collective attributes. You do this by using the `subtreeSpecification` attribute in the collective subentry.

If no `subtreeSpecification` attribute is specified in the collective subentry then the collective attributes are included in all the child entries where the collective subentry is defined.

> **See Also :**
>
> RFC 3672 "Subentries in the Lightweight Directory Access Protocol (LDAP)" at `http://www.ietf.org` for more details about the `subtreeSpecification` attribute.

The next sections provide examples of how to use the `subtreeSpecification` attribute.

- Limiting Collective Attribute to a Subtree by using base Keyword
- Controlling Number of RDNs by using minimum and maximum Keywords
- Applying Specific Exclusions by using chopBefore and chopAfter Keywords
- Including Certain Object Classes by using specificationFilter

16.1.3.1 Limiting Collective Attribute to a Subtree by using base Keyword

You use the `base` keyword in the `subtreeSpecification` to limit a collective attribute to a subtree.

For example, to restrict collective attributes only to the subtree `cn=users,dc=mycompany,dc=com`, you can use the subentry previously shown for `dc=mycompany,dc=com`, but add a `base` value to the `subtreeSpecification` attribute in the collective subentry like this:

```
SubtreeSpecification: {base "cn=users"}
```

16.1.3.2 Controlling Number of RDNs by using minimum and maximum Keywords

You use the `minimum` and `maximum` keywords in the `subtreeSpecification` attribute to control the number of RDNs from the `base` where collective attributes apply.

For example, if you want collective attributes to be added to the entries under `ou=Americas,cn=users,dc=mycompany,dc=com` but not to one level child entries of `cn=users,dc=mycompany,dc=com`, or two levels down from `cn=users, dc=mycompany,dc=com`, then define the `subtreeSpecification` as follows:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4}
```

In this configuration, `cn=john doe, ou=Americas,cn=users,dc=mycompany,dc=com` gets the collective attributes, but `cn=inbox,cn=2009,cn=emailFolder,cn=john doe, ou=Americas,cn=users,dc=mycompany,dc=com` does not get the collective attributes because it is more than four levels from the `base cn=users`.

16.1.3.3 Applying Specific Exclusions by using chopBefore and chopAfter Keywords

You can exclude collective attributes from specific entries by using the `specificExclusions`, `chopBefore`, and `chopAfter` keywords.

For example, if you do not want to add collective attributes to `ou=Europe`, define the `subtreeSpecification` as follows:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe"} }
```

If you want the collective attributes to be included in a parent DN but not its child entries, define the specification filter like this:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=Global User" } }
```

In this example, the entry `cn=Global User,ou=all region,cn=users,dc=mycompany,dc=com` gets the collective attributes, but the entry `cn=emailFolder, cn=GlobalUser,ou=all region,cn=users,dc=mycompany,dc=com` does not get the collective attributes.

16.1.3.4 Including Certain Object Classes by using specificationFilter

If you want collective attributes to be included for a certain object class only, then specify the Object Identifier or name of the object class in the `specificationFilter` for the `subtreeSpecicfication` attribute.

The following sections explain this further:

- Including a Specific Object Class
- Refining Object Classes
- Extending specificationFilter to Include a LDAP Filter

16.1.3.4.1 Including a Specific Object Class

To include the collective attributes to the `objectclass person` only, define the specification filter like this:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter
item:person }
```

Alternatively, you could use:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter
item:2.5.6.6 }
```

where `2.5.6.6` is the Object Identifier for `objectclass person`.

16.1.3.4.2 Refining Object Classes

You can use the keywords `and`, `or`, and `not` to further refine your `subtreeSpecicfication`, as follows:

```
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter and:
{ item:2.5.6.6, item:2.5.6.7} }

SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter or:
{ item:2.5.6.6, item:2.5.6.7} }

SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter not:
{ item:2.5.6.7} }
```

16.1.3.4.3 Extending specificationFilter to Include a LDAP Filter

The scope of the `specificationFilter` encompasses an LDAP filter, allowing a finer grained control of which entries to target. You can use the keyword `filter` in the subtreespecification attribute as follows:

```
subtreespecification: {base  "ou=people", filter: orclaccountstatus=enabled }
```

The following example describes how to define a filter.

```
dn: cn=building,dc=us,dc=oracle,dc=com
cn: building
subtreespecification: {base "ou=people_grp" , filter: mail=user.1@example.com }
objectclass: top
objectclass: extensibleobject
objectclass: subentry
objectclass: collectiveattributesubentry
street;collective: 200 ABC Street
```

In the preceding example, the collective attribute value `street` is only visible for entries that have `mail` attribute as `user.1@example.com`.

16.1.4 Overriding a Collective Attribute

If you want some entries to have their own values for an attribute, instead of the collective attribute value, you can specifically add that attribute to those entries and add attribute `collectiveExclusions:`*`attributeName`* to that entry.

If all the collective attributes needs to be excluded then add attribute `excludeAllCollectiveAttributes: true` to those entries. Doing so overrides the value of the collective attribute value. For example, if you add the `TelephoneNumber` attribute and the `excludeAllCollectiveAttributes: true` attribute to the entry `cn=jane smith, ou=Americas,cn=users,dc=mycompany,dc=com`, this entry will have its own value for `TelephoneNumber` instead of the collective attribute.

16.2 Managing Collective Attributes by Using the Command Line

You can manage a collective attributes sub entry from the command line, like any other directory entry.

The following sections explain this further:

- Adding a Subentry by Using ldapadd
- Modifying a Subentry by Using ldapmodify

16.2.1 Adding a Subentry by Using ldapadd

To create collective attributes, you define the collective subentry in an LDIF file using `ldapadd`.

The syntax is as follows:

```
ldapadd -p port_number -h host -D cn=orcladmin -q -f subentry.ldif
```

where the contents of `subentry.ldif` is something like this:

```
Dn: cn=collective attributes, dc=mycompany,dc=com
Cn: collective attributes
Objectclass: subentry
Objectclass: collectiveAttributeSubentry
Objectclass: top
Objectclass: extensibleobject
TelephoneNumber;collective: 1234560000
PostalCode;collective: 98765
SubtreeSpecification: {base "cn=users", minimum 2, maximum 4, specificExclusions
{ chopBefore: "ou=Europe", chopAfter: "cn=GlobalUser"},  specificationFilter not:
{ item:2.5.6.7} }
```

16.2.2 Modifying a Subentry by Using ldapmodify

To modify a subentry in a LDIF file you can use `ldapmodify` command.

Run the following command to modify the file:

```
ldapmodify -p 3060 -D "cn=orcladmin" -q -f mod_subentry.ldif
```

where `mod_subentry.ldif` is something like this:

```
dn: cn=collective attributes, dc=mycompany,dc=com
changetype: modify
replace: PostalCode;collective
PostalCode;collective: 98768
```

17

Managing Computed Attributes

You can manage computed attributes by configuring the `OrclComputedAttribute` attribute using LDAP command-line tools.
This chapter includes the following sections:

- Introduction to Computed Attributes
- Configuring Computed Attributes
- Examples of Computed Attributes Using LDAP Command-Line Tools

17.1 Introduction to Computed Attributes

The Oracle Internet Directory server provides the `OrclComputedAttribute` attribute as a mechanism to dynamically compute a configurable attribute and its value based on one or more rules. Thus, an attribute can be computed when it is actually needed, without requiring that the attribute persist in the directory store. Computed attributes can be useful in the transition from a test to a production deployment.

The `OrclComputedAttribute` attribute is a configuration attribute in the DSA Configuration entry:

```
cn=dsaconfig,cn=configsets,cn=oracle internet directory
```

`OrclComputedAttribute` is a multivalued attribute, so multiple attributes and their respective values can be computed dynamically.

An `OrclComputedAttribute` value can be derived from the following items, or a combination of these items, which are defined in the rules:

- A standard LDAP URI, as defined in RFC 4516
- A predetermined string
- An attribute value
- A function of an attribute value
- You can include the `connectBy` clause to include attributes from two or more entries. This feature uses the underlying database functionality of the SQL CONNECT BY condition with a PRIOR expression.

  For example, in an organization you can use the `connectBy` clause to include an employee's manager information, such as mail, phone numbers, and other contact information, in the employees record without actually persisting the manager's data in the employee's record.

Considerations for using `OrclComputedAttribute` include:

- The computed attribute name must be defined in the schema.
- The computed attribute might have access control policies that prevent the return of its value.

- Computed attributes are derived only for lookup and search operations. If the result entry has a configured computed attribute, then Oracle Internet Directory server skips computation of the attribute.
- Update operations are allowed for removing or replacing an existing value in the entry.

17.2 Configuring Computed Attributes

You can configure `OrclComputedAttribute` by using LDAP tools such as `ldapmodify`, Oracle Directory Services Manager (ODSM), or third-party LDAP browsers.

This section describes these topics:

- Rules and Syntax Used for Computed Attributes
- Using Special Characters With Rules for Computed Attributes

17.2.1 Rules and Syntax Used for Computed Attributes

You can use combination of rules to compute a value.

The rules for computed attributes use the following syntax to compute the attribute values:

```
OrclComputedAttribute;ComputedAttrName;dn;Filter: ldapURI

OrclComputedAttribute;ComputedAttrName;dn;Filter: "anyString"

OrclComputedAttribute;ComputedAttrName;dn;Filter: AttributeName

OrclComputedAttribute;ComputedAttrName;dn;Filter: Func(attrName)

OrclComputedAttribute;ComputedAttrName;dn;Filter: connectBy(dnAttr,Direction,Level,
[computed-attribute-rules],ldapFilter)
```

Table 17-1 describes the elements used in these rules.

If you can use a combination of these rules to compute a value, see Using Special Characters With Rules for Computed Attributes.

Table 17-1 Syntax Elements Used in Rules for Computed Attributes

Element	Description
ComputedAttrName	Name of the attribute that should be returned with the entry. The computed attribute name must be defined in the schema. The attribute is also subject to ACL evaluation after the value is computed.
dn	Distinguished name. The attribute is computed for child entries under this DN.
Filter	Filter value. The attribute is computed for entries that belong to this filter value.
ldapURI	URI that conforms to the syntax described in RFC 4516, as follows: `ldap:///baseDN?ReqdAttribute??scope?filter` *ReqdAttribute* is a single attribute name for a required attribute. If there is a space character in the *ldapURI* configuration, the space must be encoded as `%20`, as described in RFC 4516.
"*anyString*"	String to be included. It must be enclosed by double quote characters.
AttributeName	Attribute name that indicates the value of that attribute should be used.

Table 17-1 (Cont.) Syntax Elements Used in Rules for Computed Attributes

Element	Description
`Func(attr)`	Name of a function to perform on the value. The following functions are available: `lower`(*attrName*) `upper`(*attrName*) `substr`(*attrName*,*pos*,*len*) `replace`(*attrName*,*"str"*) `trunc`(*attrName*, *"c"*) `NVL`(*attr*, *"StringVal"*) **Note:** In case of the `NVL` function, if the *attr* attribute is missing in the entry, then the Oracle Internet Directory server generates the computed attribute with the default string value, *StringVal*.
`connectBy`	The `connectBy` clause uses the following syntax: `connectBy(`*`dnAttr`*`,`*`Direction`*`,`*`Level`*`,[`*`computed-attribute-rules`*`],`*`ldapFilter`*`)` • *`dnAttr`* is the DN syntax for the attribute in the target entry. For example: `manager`, `owner`, or `uniquemember`. It is a mandatory parameter. • *`Direction`* is the direction in the directory to return results: 0 - downwards 1 - upwards It is a mandatory parameter. • *`Level`* is the number of recursion levels downwards or upwards in the directory to return results. It is a mandatory parameter. • *`computed-attribute-rules`* are applied to each entry that is fetched as result of the `connectBy` clause. These rules are enclosed in square brackets []. It is an optional parameter. • *`ldapFilter`* causes the computed attribute to be generated only if the filter condition matches on the target entry. It is an optional parameter.

17.2.2 Using Special Characters With Rules for Computed Attributes

You can use special characters for computed attributes.

Considerations for using special characters with the rules for computed attributes include:

- If you specify more than one rule, you must use a space character to separate each rule.
- To append or concatenate a rule or string, use a plus sign (+).
- To have the result of the first rule evaluation determine the value, use the OR operator (|).
- The asterisk (*) is the wildcard character and is allowed only in *`ldapURI`*. If this operator appears in the DN part of the URI, then * is derived from the scope of evaluating the entry's DN.

17.3 Examples of Computed Attributes Using LDAP Command-Line Tools

LDAP command-line tool is used to return the attributes with uppercase, to replace an attribute value etc.

This section provides the following examples for computed attributes:

- Returning an Attribute Value as Uppercase
- Returning the Substring of an Attribute Value
- Replacing an Attribute Value
- Specifying a URI-Based Configuration
- Using a Combination of Different Rules
- Using an OR (|) Operator
- Using the `connectBy` Interface
- Creating Hierarchical Groups Using `connectBy`

17.3.1 Returning an Attribute Value as Uppercase

You can use `upperattr` to return an attribute value as uppercase.

This example computes the `cn` attribute as uppercase and returns the attribute as `upperattr`:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;upperattr;dc=acme,dc=com;objectclass=person
orclcomputedattribute;upperattr;ou=EuroSInet Suite,o=IMC,c=US;objectclass=person:
upper(cn)
```

The schema definition for `upperattr` must be defined.

17.3.2 Returning the Substring of an Attribute Value

You can compute the substring of an attribute by using `substrattr`.

This example computes the substring of the attribute value and returns the attribute `substrattr` with the value as description value from position 1 for the next 3 characters:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;substrattr;dc=acme,dc=com;objectclass=person
orclcomputedattribute;substrattr;dc=acme,dc=com;objectclass=person: substr(description,
1,3)
```

17.3.3 Replacing an Attribute Value

You can replace an attribute value by using `replace` command.

This example computes and replaces the attribute value for `newTitle`:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;newTitle;cn=john doe,dc=acme,dc=com;objectclass=person
orclcomputedattribute;newTitle;ou=EuroSInet Suite,o=IMC,c=US;objectclass=person:
replace(title, "Clerk", "Manager")
```

17.3.4 Specifying a URI-Based Configuration

You can specify URI-based configuration.

This example adds `commonTelephoneNumber` to every entry under `dc=acme,dc=com`:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;commonTelephoneNumber;dc=acme,dc=com;objectclass=person
orclcomputedattribute;commonTelephoneNumber;dc=acme,dc=com,c=US;objectclass=person:
"Common PhoneNumber is " +
ldap:///cn=common%20attributes,dc=com?telephonenumber??base?objectclass=*
```

17.3.5 Using a Combination of Different Rules

You can combine different rules using special characters.

Concatenate different rules using a + (plus sign) with the rules separated by a space character:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;combinationAttribute;ou=EuroSInet
Suite,o=IMC,c=US;objectclass=person
orclcomputedattribute;combinationAttribute;dc=acme,dc=com;objectclass=person: "telephone
number from common entry:" +  ldap:///cn=common Entry,?telephonenumber??base?
objectclass=* +
" appending replace of title attr with clerk/manager " +
replace(title, "Clerk", "Manager")

dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;combinationAttr2;dc=acme,dc=com;objectclass=person
orclcomputedattribute;combinationAttr2;dc=acme,dc=com;objectclass=person: "Telephone
number  from common entry " +
ldap:///cn=commonEntry,dc=acme,dc=com?telephonenumber??base?objectclass=* +
" appending truncate of description for space char " + trunc(description, " ")
```

17.3.6 Using an OR (|) Operator

The OR operator (|) is used to get the output by checking two conditions.

Add the `contactNumber` attribute as `telephoneNumber` if the entry has the `telephoneNumber` attribute; otherwise, copy the value 6505067000:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;contactNumber;cn=employees,dc=acme,dc=com;(objectclass=person)
orclcomputedattribute;contactNumber;cn=employees,dc=acme,dc=com;(objectclass=person):
telephonenumber | 6505067000
```

> **Note:**
>
> For the OR operator (|), the computed attribute definition supports the following syntax:
>
> ```
> definition-1 | definition-2 | ... definition-n
> ```
>
> where:
>
> - definition-1 can be a complex rule.
> - definition-2 through definition-*n* must be a simple rule only, such as a hard-coded string or an attribute value. These definitions cannot be function expressions or a combination of expression rules.
>
> For example, for the following rule, if the uid is not present for an entry, the result will be a computed value "cn=", regardless of the number of occurrences of the attribute cn in the entry:
>
> ```
> orclcomputedattr;myattr;cn=employees;(objectclass=inetorgperson): "uid=" + uid
> | "cn=" + cn
> ```

17.3.7 Using the `connectBy` Interface

The example describes the value from the attribute (`manager`) of a target entry and then generates the `AllReports` computed attributes for that manager's reports for up to 10 levels downwards in the directory.

The manager's DN is:

```
dn: uid=Manager,ou=people,dc=us,dc=example,dc=com

dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;AllReports;dc=com;objectclass=person
orclcomputedattribute;AllReports;dc=com;objectclass=person:connectby(manager,0,10)
```

Each computed attribute includes the respective employee's `uid`. For example:

```
AllReports=uid=employee1,ou=people,dc=us,dc=example,dc=com
...
AllReports=uid=employee6,ou=people,dc=us,dc=example,dc=com
```

The following example generates the `AllManagers` computed attribute for each employee up to 15 levels upwards in the directory. The DN is:

```
dn: uid=manager,ou=people,dc=us,dc=example,dc=com

dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;AllManagers;dc=com;objectclass=person
orclcomputedattribute;AllManagers;dc=com;objectclass=person: connectBy(manager,1,15,
[ "EmpNum " + employeenumber + " " + upper(orclnormdn) ])
```

Each computed attribute includes the respective manager's `employeenumber` and normalized DN of the entry (`orclnormdn` attribute) in uppercase. For example:

```
AllManagers=EmpNum1 UID=NAME1,OU=PEOPLE,DC=US,DC=EXAMPLE,DC=COM
...
AllManagers=EmpNum5 UID=NAME5,OU=PEOPLE,DC=US,DC=EXAMPLE,DC=COM
```

The following example shows the preceding example with the filter `(objectclass=inetorgperson)` added:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;AllManagers;dc=com;objectclass=person
orclcomputedattribute;AllManagers;dc=com;objectclass=person:connectBy(manager,1,15,
[ "EmpNum " + employeenumber + " " + upper(orclnormdn) ],
(objectclass=inetorgperson))
```

17.3.8 Creating Hierarchical Groups Using `connectBy`

`connectBy` is used to create hierarchical groups.

The following example shows the `connectBy` clause for a recursive (hierarchical) group.

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclcomputedattribute;uniquemember;dc=com;objectclass=groupOfUniquenames
orclcomputedattribute;uniquemember;dc=com;objectclass=groupOfUniquenames:connectBy(unique
member,1,25, [uniquemember])
```

The following example shows a static and a hierarchical group computed by Oracle Internet Directory server. Two static group entries that contain the direct reports of the manager are added.

Given the static unique members stored in the group entry, the example returns nested results going upwards in the directory of `uniquemember` attributes. For example:

```
dn: cn=manager1_org,cn=groups,dc=us,dc=example,dc=com
objectclass: groupofuniquenames
objectclass: top
cn: manager1_org
uniquemember: uid=manager1,ou=People,dc=us,dc=example,dc=com
uniquemember: uid=emp1,ou=People,dc=us,dc=example,dc=com
uniquemember: uid=emp2,ou=People,dc=us,dc=example,dc=com
uniquemember: uid=emp3,ou=People,dc=us,dc=example,dc=com
uniquemember: uid=emp4,ou=People,dc=us,dc=example,dc=com
uniquemember: cn=emp2_org,cn=groups,dc=us,dc=example,dc=com
```

and

```
dn: cn=emp2_org,
cn=groups,dc=us,dc=example,dc=com
objectclass: groupofuniquenames
objectclass: top
cn: emp2_org
uniquemember: uid=emp5,ou=People,dc=us,dc=example,dc=com
uniquemember: uid=emp6,ou=People,dc=us,dc=example,dc=com
```

When the entry "`cn=manager1_org,cn=groups,dc=us,dc=example,dc=com`" is searched, Oracle Internet Server server automatically computes `uniquemember` attributes recursively. For example:

```
cn=manager1,cn=groups,dc=us,dc=example,dc=com
uniquemember=cn=emp2_org,cn=groups,dc=us,dc=example,dc=com
uniquemember=uid=manager1,ou=people,dc=us,dc=example,dc=com
```

```
uniquemember=uid=emp1,ou=people,dc=us,dc=example,dc=com
uniquemember=uid=emp2,ou=people,dc=us,dc=example,dc=com
uniquemember=uid=emp3,ou=people,dc=us,dc=example,dc=com
uniquemember=uid=emp4,ou=people,dc=us,dc=example,dc=com
uniquemember=uid=emp5,ou=people,dc=us,dc=example,dc=com
uniquemember=uid=emp6,ou=people,dc=us,dc=example,dc=com
```

18

Managing Alias Entries

This chapter describes Oracle Internet Directory alias entries and how to add, search for, and modify alias entries using LDAP command-line tools. It also explains how to interpret messages related to alias dereferencing.
This chapter includes the following sections:

- Introduction to Alias Entries
- Adding an Alias Entry
- Searching the Directory with Alias Entries
- Modifying Alias Entries
- Messages Related to Alias Dereferencing

For more information about attribute aliases, see Understanding Attribute Aliases .

18.1 Introduction to Alias Entries

Entries sometimes have distinguished names that are long and cumbersome. Oracle Internet Directory makes it easier to administer long names by using alias objects. When someone looks up—that is, references—an object by using an alias, the alias is dereferenced, and what is returned is the object to which the alias points.

For example, the alias, `Server1`, can be dereferenced so that it points to the fully qualified DN namely, `dc=server1,dc=us,dc=myCompany,dc=com`. This feature also enables you to devise structures that are not strictly hierarchical.

An alias entry uses the object class `alias` to distinguish it from object entries in a directory. The definition of that object class is as follows:

```
(2.5.6.1 NAME 'alias' SUP top STRUCTURAL MUST aliasedObjectName)
```

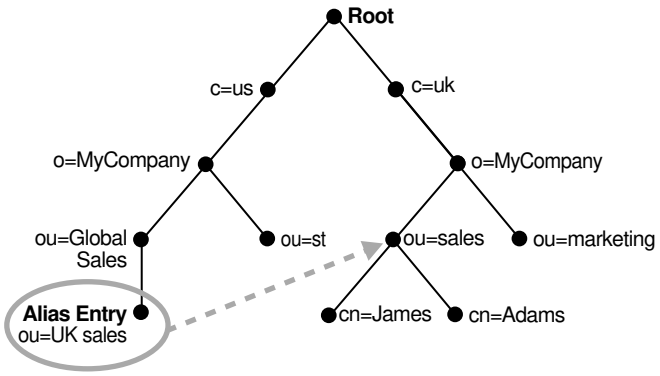
An alias entry also contains the `aliasedObjectName` attribute that, in turn, contains the DN of the object to which it is pointing. The definition of that attribute is as follows:

```
(2.4.5.1 NAME 'aliasedObjectName" EQUALITY distinguishedNmameMatch SYNTAX
1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE)
```

> **Note:**
>
> The `aliasedObjectName` attribute should not refer to the Directory Information Tree (DIT) view DN.

Figure 18-1 and the accompanying text provides an example of alias entry dereferencing.

Figure 18-1 Alias Entries Example



In Figure 18-1, `ou=uk sales,ou=global sales,o=myCompany,c=us` is an alias entry pointing to the `ou=sales,o=myCompany,c=uk` entry.

When anyone references `ou=uk sales,ou=global sales,o=oracle,c=us`, the directory server automatically reroutes them to the real entry, `ou=sales,o=oracle,c=uk`.

18.2 Adding an Alias Entry

To add an alias entry, you create a normal entry in LDIF and an alias entry pointing to the real entry.

Following the steps in this example produces the tree in Figure 18-2.

1. Create a sample LDIF file, My_file.ldif, with the following entries:

```
dn: c=us
c: us
objectclass: country

dn: o=MyCompany, c=us
o: MyCompany
objectclass:organization

dn: ou=Area1, c=us
objectclass: alias
objectclass: extensibleobject
ou: Area1
aliasedObjectName: o=MyCompany, c=us

dn: cn=John Doe, o=MyCompany, c=us
cn: John Doe
sn: Doe
objectclass: person

dn: cn=President, o=MyCompany, c=us
objectclass: alias
objectclass: extensibleobject
cn: President
aliasedobjectname: cn=John Doe, o=MyCompany, c=us
```

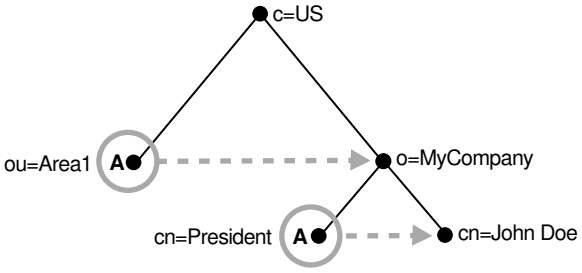
2. Add these entries to the directory by using the following command:

```
ldapadd -p port -h host -D cn=orcladmin -q -f My_file.ldif
```

> **Note:**
>
> If you attempt to add an alias entry whose parent is an alias entry, the directory server returns an error.

> **See Also:**
>
> Table 18-2 for error messages.

Figure 18-2 Resulting Tree when Creating the My_file.ldif



In Figure 18-2, the letter A represents an alias entry, where:

- `ou=Area1` is an alias pointing to `o=MyCompany`
- `cn=President` is an alias pointing to `cn=John Doe`

18.3 Searching the Directory with Alias Entries

You can search the base, one-level down the base, and a subtree by using flags.

The following sections explain this further:

- Flags for Searching the Directory with Alias Entries
- Searching the Base with Alias Entries
- Searching One-Level with Alias Entries
- Searching a Subtree with Alias Entries

18.3.1 Flags for Searching the Directory with Alias Entries

In each search you specify, there are flags you can set. The search is performed based on the flag you specify.

See Table 18-1for directory search behaviour based on alias flags.

Table 18-1 Flags for Searching the Directory with Alias Entries

Flag	Search Behavior of LDAP Server
`-a never`	Never dereferences aliases.
`-a find`	Dereferences the base object in a search, but does not dereference alias entries that are under the base.
`-a search`	Dereferences aliases in subordinates of the base object in search but not in locating the base object of the search.
`-a always`	Dereferences aliases both in searching and in locating the base object of the search.

By default, the dereference flag in `ldapsearch` is `-a never` and thus the directory server does not perform any dereferencing for alias entries.

18.3.2 Searching the Base with Alias Entries

A base search finds the top level of the alias entry you specify.

The following sections explain this further:

- Searching the Base with Dereferencing Flag -a find
- Searching the Base with Dereferencing Flag -a search
- Searching the Base with Dereferencing Flag -a always

18.3.2.1 Searching the Base with Dereferencing Flag -a find

This example shows a base search of `ou=Area1,c=us` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a find`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a find -s base "objectclass=*"
```

The directory server, during the base search, looks up the base specified in the search request and returns it to the user. However, if the base is an alias entry and, as in the example, `-a find` is specified in the search request, then the directory server automatically dereferences the alias entry and returns the entry it points to. In this example, the search dereferences `ou=Area1,c=us`, which is an alias entry, and returns `o=MyCompany,c=us`.

18.3.2.2 Searching the Base with Dereferencing Flag -a search

This example shows a base search of `ou=Area1,c=us` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a search`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a search -s base "objectclass=*"
```

The directory server, during the base search, looks up the base specified in the search request and returns it to the user without dereferencing it. It returns `ou=Area1,c=us`.

18.3.2.3 Searching the Base with Dereferencing Flag -a always

This example shows a base search of `ou=Area1,c=us` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a always`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a always -s base "objectclass=*"
```

The directory server, during the base search, looks up the base specified in the search request. If it is an alias entry, the directory server automatically dereferences the alias entry and returns the entry it points to. In this example, the search dereferences `ou=Area1,c=us`, which is an alias entry, and returns `o=MyCompany,c=us`.

18.3.3 Searching One-Level with Alias Entries

A one-level search finds only the children of the base level you specify.

The following sections explain this further:

- Searching One-Level with Dereferencing Flag -a find
- Searching One-Level with Dereferencing Flag -a search
- Searching One-Level with Dereferencing Flag -a always

18.3.3.1 Searching One-Level with Dereferencing Flag -a find

This example shows a one-level search of `"ou=Area1,c=us"` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a find`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a find -s one "objectclass=*"
```

The directory server returns one-level entries under the base that match the filter criteria. In the example, `-a find` is specified in the search request, and thus the directory server automatically dereferences while looking up the base (the first step), but does not dereference alias entries that are one level under the base. Therefore, the search dereferences `ou=Area1,c=us`, which is an alias entry, and then looks up one-level entries under `o=MyCompany,c=us`. One of the one-level entries is `cn=President,o=MyCompany,c=us` that is not dereferenced and is returned as is.

Thus, the search returns `cn=President,o=MyCompany,c=us` and `cn=John Doe,o=MyCompany,c=us`.

18.3.3.2 Searching One-Level with Dereferencing Flag -a search

This example shows a one-level search of `"ou=Area1,c=us"` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a search`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a search -s one "objectclass=*"
```

The directory server searches for the base that is specified in the search request. If the base entry is an alias entry, it returns nothing. (Alias entries cannot have children.) Otherwise, it returns the base entry's immediate children after dereferencing them. In this example, the base entry is `"ou=Area1,c=us"`, which is an alias entry, so the search returns nothing

18.3.3.3 Searching One-Level with Dereferencing Flag -a always

This example shows a one-level search of `"ou=Area1,c=us"` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a always`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a always -s one "objectclass=*"
```

In the example, `-a always` is specified in the search request, and thus the directory server automatically dereferences while looking up the base (the first step), then dereference alias entries that are one level under the base. Therefore, the search dereferences `ou=Area1,c=us`,

which is an alias entry, and then looks up one-level entries under `o=MyCompany,c=us`. One of the one-level entries is `cn=President,o=MyCompany,c=us`. That is dereferenced and is returned as `cn=John Doe,o=MyCompany,c=us`. The other one-level entry is `cn=John Doe,o=MyCompany,c=us`, which has already been returned.

Thus, the search returns `cn=John Doe,o=MyCompany,c=us`.

18.3.4 Searching a Subtree with Alias Entries

A subtree search finds the base, children, and grand children.

The following sections explain this further:

- Searching Subtree with Dereferencing Flag -a find
- Searching Subtree with Dereferencing Flag -a search
- Searching Subtree with Dereferencing Flag -a always

18.3.4.1 Searching Subtree with Dereferencing Flag -a find

This example shows a subtree search of `"ou=Area1,c=us"` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a find`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a find -s sub "objectclass=*"
```

The directory server returns all entries under the base that match the filter criteria. In the example, `-a find` is specified in the search request, and thus the directory server automatically dereferences while looking up the base (the first step), but does not dereference alias entries that are under the base. Therefore, the search dereferences `ou=Area1,c=us`, which is an alias entry, and then looks up entries under `o=MyCompany,c=us`. One of the entries is `cn=President,o=MyCompany,c=us` that is not dereferenced and is returned as is.

Thus, the search returns:

- `o=MyCompany,c=us`
- `cn=John doe,o=MyCompany,c=us`
- `cn=President,o=MyCompany,c=us`

18.3.4.2 Searching Subtree with Dereferencing Flag -a search

This example shows a subtree search of `"ou=Area1,c=us"` with a filter of `"objectclass=*"` with the dereferencing flag set to `-a search`.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a search -s sub "objectclass=*"
```

The directory searches for the base that is specified in the search request.If the base is an alias entry, then it returns the base entry without dereferencing it. (Alias entries cannot have children.) Otherwise it returns all entries under the base. If any alias entries are found, it dereferences them and returns all entries under them as well.

In this example, the base entry is an alias entry, `ou=Area1,c=us`, so the directory returns `ou=Area1,c=us`.

18.3.4.3 Searching Subtree with Dereferencing Flag -a always

This example shows a subtree search of "ou=Area1,c=us" with a filter of "objectclass=*" with the dereferencing flag set to -a always.

```
ldapsearch -p port -h host -b "ou=Area1,c=us" -a always -s sub "objectclass=*"
```

The directory server dereferences the base entry and returns it. It also returns all entries under the dereferenced base. If any alias entries are found, it dereferences them and returns all entries under them as well.

In this example, the base entry is ou=Area1,c=us, which is dereferenced to o=MyCompany,c=us, which is returned. There are two entries under o=MyCompany,c=us. One is cn=President,o=MyCompany,c=us, which is returned and also dereferenced to cn=John Doe,o=MyCompany,c=us, which is returned. The other entry under o=MyCompany,c=us, which has already been returned. So the result is o=MyCompany,c=us and cn=John Doe,o=MyCompany,c=us.

18.4 Modifying Alias Entries

You can modify alias entries using ldapmodify command.

This example shows how to modify alias entries. It creates a sample LDIF file, My_file.ldif with following entries:

```
dn: cn=President, o=MyCompany, c=us
changetype : modify
replace: aliasedobjectname
aliasedobjectname: cn=XYZ, o=MyCompany, c=us
```

Modify the alias entry using the following command:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -f My_file.ldif
```

18.5 Messages Related to Alias Dereferencing

A dereferencing message is displayed when there is a problem associated with aliases.

Table 18-2 lists the messages related to alias entry dereferencing and the corresponding meaning for each message.

Table 18-2 Entry Alias Dereferencing Messages

Message	Meaning
Alias Problem	Either of the following have occurred: • An alias was dereferenced, but it did not point to an entry in the DIT. • The user tries to add an alias entry whose parent is an alias.
Alias Dereferencing Problem	The user cannot dereference an alias because of access control issues.
No Such Object	The server cannot find the base DN specified in the search request.
Invalid DN Syntax	When adding or modifying an alias entry, if the value specified for aliasedObjectName has invalid DN syntax, then the directory server returns this error message to the client.

Table 18-2 (Cont.) Entry Alias Dereferencing Messages

Message	Meaning
Success	The client operation successfully completes. When the dereferenced target does exist but does not match the filter specified in the search request, the server returns a success message with no matched entry.
Insufficient Access Rights	The user does not have access to the dereferenced entry.

19

Managing Attribute Uniqueness Constraint Entries

You can define attribute uniqueness in Oracle Internet Directory and clean up duplicate attribute values, specify attribute uniqueness constraints, and manage attribute uniqueness entries using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities. This chapter includes the following sections:

- Introduction to Managing Attribute Uniqueness Constraint Entries
- Duplicate Attribute Values
- Cleaning Up Duplicate Attribute Values
- Specifying Attribute Uniqueness Constraint Entries
- Managing an Attribute Uniqueness Constraint Entry by Using ODSM
- Managing Attribute Uniqueness Constraint Entries by Using the Command Line

19.1 Introduction to Managing Attribute Uniqueness Constraint Entries

You can define attribute uniqueness across an entire directory, across one subtree, and across one object class. You can implement attribute uniqueness by creating an attribute uniqueness constraint entry.

The following sections explain this further:

- Attribute Uniqueness Scope
- Attribute Uniqueness Constraint Entries
- Attribute Uniqueness Constraint in Oracle Internet Directory Replication Environment
- Support of LDAP Tools for Attribute Uniqueness

19.1.1 Attribute Uniqueness Scope

You can define attribute uniqueness across an entire directory, across one subtree, and across one object class.

- Across the entire directory

  For example, to ensure that every entry in your directory that includes a `mail` attribute has a unique value for that attribute, you create an instance of attribute uniqueness associated with `mail`.

- Across one subtree for each attribute

  For example, suppose that MyCompany hosts the directories for SubscriberCompany1 and SubscriberCompany2. You can choose to enforce attribute uniqueness in SubscriberCompany1 only.

- Across one object class

  For example, suppose that `ID` is an attribute in both the `machine` object class and the `person` object class. If attribute uniqueness is enabled, then the directory server prevents you from adding either two machines or two people with the same `ID`. You can, however, add a `machine ID` attribute that has the same value as an existing `person ID` attribute. Similarly, you can add a `person ID` attribute that has the same value as an existing `machine ID` attribute.

19.1.2 Attribute Uniqueness Constraint Entries

Attribute uniqueness constraint entries are stored under `cn=unique,cn=Common,cn=Products,cn=OracleContext`.

To implement attribute uniqueness, you create an attribute uniqueness constraint entry in which you provide values for the attributes shown in Table 19-1.

Table 19-1 Attribute Uniqueness Constraint Entry

Attribute Name	Mandatory?	Valid Value	Default Value	Default Effect
`orcluniqueattrname`	Yes	Any string	N/A	N/A
`orcluniquescope`	No	One of the following: • `base`: Searches the root entry only • `onelevel`: Searches one level only • `sub`: Searches the entire directory	`sub`	Searches the entire directory
`orcluniqueenable`	No	Either `0` (disable) or `1` (enable)	`0`	Disables attribute uniqueness
`orcluniquesubtree`	No	Any string	" "	Searches the entire directory
`orcluniqueobjectclass`	No	Any string	" "	Searches all object classes

19.1.3 Attribute Uniqueness Constraint in Oracle Internet Directory Replication Environment

When an attribute uniqueness constraint is present in the Oracle Internet Directory replication environment, be careful about configuring the attribute uniqueness constraints on each server.

This section contains these topics:

- Attribute Uniqueness Constraint in Simple Replication Environment
- Attribute Uniqueness Constraint in Multimaster Replication Environment

19.1.3.1 Attribute Uniqueness Constraint in Simple Replication Environment

When attribute uniqueness constraint is present in Oracle Internet Directory simple replication environment:

Because all modifications by client applications are performed on the supplier server, the attribute uniqueness constraint should be enabled on that server. It is not necessary to enable

the attribute uniqueness constraint on the consumer server. Enabling the attribute uniqueness constraint on the consumer server does not prevent the directory server from operating correctly, but it can cause a performance degradation.

19.1.3.2 Attribute Uniqueness Constraint in Multimaster Replication Environment

When attribute uniqueness constraint is present in Oracle Internet Directory multimaster replication environment:

In a multimaster replication scenario, nodes serve as both suppliers and consumers of the same replica. Multimaster replication uses a loosely consistent replication model.

Enabling an attribute uniqueness constraint on one of the servers does not ensure that attribute values are unique across both masters at any given time. Enabling an attribute uniqueness constraint on only one server can cause inconsistencies in the data held on each replica.

The attribute uniqueness constraint must be enabled on both masters. However, there may still be an inconsistent state. For example, in both masters we can successfully modify entries to the same attribute value. However, when the changes are later replicated to the other node, the conflict becomes apparent. You must take this type of conflict resolution into consideration as well, deciding whether conflict resolution should be the replication server's responsibility.

19.1.4 Support of LDAP Tools for Attribute Uniqueness

When you use the LDAP tools, the attribute uniqueness feature prevents duplication of attribute values, both when adding and modifying them. For example, it prevents you from assigning to a new employee an identifier already assigned to another employee. Instead, the directory server terminates the operation and returns an error message.

> **Note:**
>
> The LDAP tools support attribute uniqueness. The bulk tools does not support attribute uniqueness.

When you have created the entry and specified the attributes, before it performs an operation, the directory server:

- Uses the attribute uniqueness constraint to check all update operations
- Determines whether the operation applies to a monitored attribute, subtree, or object class

If an operation applies to a monitored attribute, suffix, or object class, and would cause two entries to have the same attribute value, then the directory server terminates the operation and returns a constraint violation error message to the client.

> **Note:**
>
> The attribute uniqueness feature works on indexed attributes only.

19.2 Duplicate Attribute Values

In earlier releases, if duplicate attribute values existed in the directory before attribute uniqueness was enabled, Oracle Internet Directory server did not report an error for these duplicate values.

During an upgrade to Oracle Internet Directory 14c (14.1.2.1.0), data is copied to the `attr_uniqueness` table. During this upgrade process, duplicate values are copied the `attr_uniqueness_log` table and not to the `attr_uniqueness` table.

After a fresh installation of Oracle Internet Directory, when the attribute uniqueness is enabled existing data can also have duplicate values.

19.3 Cleaning Up Duplicate Attribute Values

You need to determine and clean up duplicate attribute values.

Follow these steps:

1. Get the list of duplicate attribute values using SQL*Plus. For example:

   ```
   sqlplus ods@oiddb
   spool duplicateVals.out
   select attrvalue from attr_uniqueness_log ;
   spool off
   quit
   ```

   When prompted, enter your Oracle Database password.

2. For each duplicate value in `duplicateVals.out`, take an appropriate action, depending on your deployment.

   To find entries with duplicate values, use `ldapsearch`. For example:

   ```
   ldapsearch -p -D cn=orcladmin -q -b " " -s sub "attributeName=duplicateValue"
   ```

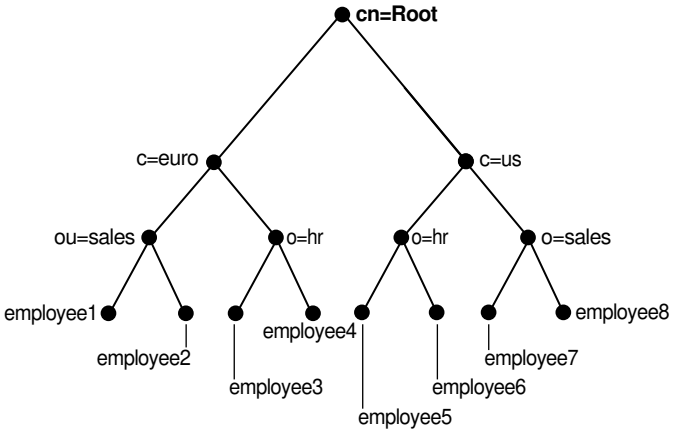
   For example, you can remove the duplicate value from the entry, replace the value with a new value, or delete the entry altogether.

19.4 Specifying Attribute Uniqueness Constraint Entries

Attribute uniqueness constraint entries are stored under `cn=unique,cn=Common,cn=Products,cn=OracleContext`.

To understand the examples in this section, refer to Figure 19-1.

Figure 19-1 Example of a Directory Information Tree



This section describes and gives examples of rules you follow when creating attribute uniqueness constraints. It contains these topics:

- Specifying Multiple Attribute Names in an Attribute Uniqueness Constraint
- Specifying Multiple Subtrees in an Attribute Uniqueness Constraint
- Specifying Multiple Scopes in an Attribute Uniqueness Constraint
- Specifying Multiple Object Classes in an Attribute Uniqueness Constraint
- Specifying Multiple Subtrees, Scopes, and Object Classes in an Attribute Uniqueness Constraint

19.4.1 Specifying Multiple Attribute Names in an Attribute Uniqueness Constraint

When multiple attribute uniqueness constraints have different values in `orcluniqueattrname`, their effects are independent of each other.

For example, suppose that a user defines two attribute uniqueness constraints as follows:

Constraint1:

```
orcluniqueattrname: employee_id
```

Constraint2:

```
orcluniqueattrname: email_id
```

In this example, Constraint1 and Constraint2 enforce uniqueness on the specified attribute within their own attribute uniqueness scopes. Constraint1 and Constraint2 are independent of each other.

19.4.2 Specifying Multiple Subtrees in an Attribute Uniqueness Constraint

When multiple attribute uniqueness constraints have the same values in `orcluniqueattrname`, `orcluniquescope` and `orcluniqueobjectclass`, but different values in `orcluniquesubtree`, the subtree scopes specified by those attribute uniqueness constraints are checked individually.

For example, refer to Figure 19-1. Suppose that a user defines two attribute uniqueness constraints as follows:

Constraint1:

```
orcluniqueattrname: employee_id
orcluniquesubtree: o=sales, c=us, cn=root
orcluniquescope: onelevel
```

Constraint2:

```
orcluniqueattrname: employee_id
orcluniquesubtree: o=hr, c=euro, cn=root
orcluniquescope: onelevel
```

In this example, the attribute uniqueness on `employee_id` is enforced against all entries under subtree `o=sales,c=us,cn=root`. Attribute uniqueness on `employee_id` is also enforced against all entries under `o=hr,c=euro,cn=root` independent of the entries under the subtree `o=sales,c=us,cn=root`—that is, the directory server enforces the unique value of the `employee_id` attribute for employee3 and employee4. Unique `employee_id` is enforced for employee7 and employee8 as well while employee7 could have the same `employee_id` as employee4.

19.4.3 Specifying Multiple Scopes in an Attribute Uniqueness Constraint

When multiple attribute uniqueness constraints have the same values in `orcluniqueattrname`, `orcluniquesubtree` and `orcluniqueobjectclass`, but different values in `orcluniquescope`, the attribute uniqueness constraint with the largest search scope takes effect.

For example, referring to Figure 19-1, suppose that a user defines two attribute uniqueness constraints as follows:

Constraint1:

```
orcluniqueattrname: employee_id
orcluniquesubtree: c=us, cn=root
orcluniquescope: onelevel
```

Constraint2:

```
orcluniqueattrname: employee_id
orcluniquesubtree: c=us, cn=root
orcluniquescope: sub
```

In this example, the attribute uniqueness on `employee_id` is enforced against all entries under the subtree `c=us,cn=root` and the entry `c=us,cn=root` itself. Note that this is the same as if the user had defined only `Constraint2`.

19.4.4 Specifying Multiple Object Classes in an Attribute Uniqueness Constraint

When multiple attribute uniqueness constraints have the same values in `orcluniqueattrname`, `orcluniquesubtree`, and `orcluniquescope`, but different values in `orcluniqueobjectclass`, then the union of attributes belonging to those object classes is checked.

For example, look at Figure 19-1. Suppose that a user defines two attribute uniqueness constraints as follows:

Constraint1:

```
orcluniqueattrname: employee_id
orcluniquesubtree: c=us, cn=root
orcluniqueobjectclass: person
```

Constraint2:

```
orcluniqueattrname: employee_id
orcluniquesubtree: c=us, cn=root
```

In this example, the attribute uniqueness on `employee_id` is enforced against all entries under the subtree `c=us,cn=root` and the entry `c=us,cn=root` itself, no matter what object class those entries belong to. Note that Constraint2 specifies no `orcluniqueobjectclass` attribute, which is the same as specifying all object classes.

19.4.5 Specifying Multiple Subtrees, Scopes, and Object Classes in an Attribute Uniqueness Constraint

When multiple attribute uniqueness constraints have the same values in `orcluniqueattrname`, but different values in `orcluniquesubtree`, `orcluniquescope`, and `orcluniqueobjectclass`, the entries that belong to the attribute uniqueness scopes of different constraints are checked individually.

For example, referring to Figure 19-1, suppose that a user defines two attribute uniqueness constraints as follows:

Constraint1:

```
orcluniqueattrname: employee_id
orcluniquesubtree: o=sales, c=us, cn=root
orcluniquescope: onelevel
orcluniqueobjectclass: person
```

Constraint2:

```
orcluniqueattrname: employee_id
orcluniquesubtree: c=euro, cn=root
orcluniquescope: sub
orcluniqueobjectclass: organization
```

In this example, the attribute uniqueness on `employee_id` is enforced against each of the following independent of each other:

- All entries one level under the entry `o=sales,c=us,cn=root` with the object class `person`

- All entries under subtree `c=euro,cn=root` and the entry `c=euro,cn=root` itself with the object class `organization`

19.5 Managing an Attribute Uniqueness Constraint Entry by Using ODSM

You can manage an attribute uniqueness constraint policy by using Oracle Directory Services Manager (ODSM).

The following sections explain this further:

- Creating an Attribute Uniqueness Constraint Entry by Using ODSM
- Modifying an Attribute Uniqueness Constraint Entry by Using ODSM
- Deleting an Attribute Uniqueness Constraint Entry by Using ODSM

19.5.1 Creating an Attribute Uniqueness Constraint Entry by Using ODSM

Using ODSM, you can create an attribute uniqueness constraint entry.

To create an attribute uniqueness constraint entry by using ODSM:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. Select **Advanced** from the task selection bar.
3. Expand **Attribute Uniqueness** in the left pane.
4. On the toolbar, choose the **Create an attribute uniqueness constraint** icon. This displays the New Constraint window.
5. In the New Constraint dialog box, enter values in the text fields and select the **Unique Attribute Scope**. You can click **Browse** to select the Unique Attribute Subtree.
6. If you want to enable the constraint now, click Enable **Unique Attribute.**
7. Choose **OK**. The entry you just created appears in the list of attribute uniqueness constraint entries in the left panel.
8. Click **Apply** to apply this constraint or **Revert** to revert to the state before you created the new entry.

19.5.2 Modifying an Attribute Uniqueness Constraint Entry by Using ODSM

Using ODSM, you can modify an attribute uniqueness constraint.

To modify an attribute uniqueness constraint entry by using ODSM:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. Select **Advanced** from the task selection bar.
3. Expand **Attribute Uniqueness** in the left pane.
4. Select an existing uniqueness constraint. This displays the General tab of the Attribute Uniqueness Constraint window.
5. Enter or modify values.

6. If you want to enable the constraint now, click Enable **Unique Attribute.**
7. Click **Apply** to apply this change or **Revert** to revert to the state before you modified the entry.

19.5.3 Deleting an Attribute Uniqueness Constraint Entry by Using ODSM

You can delete an attribute uniqueness constraint entry by using ODSM.

To delete an attribute uniqueness constraint policy:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. Select **Advanced** from the task selection bar.
3. Expand **Attribute Uniqueness** in the left pane.
4. In the left panel, select the attribute uniqueness constraint entry you want to delete.
5. Choose the **Delete** icon, then, when prompted, confirm the deletion. The entry you deleted no longer appears in the list of attribute uniqueness constraint entries in the left panel.
6. Click **Apply** to apply this change or **Revert** to revert to the state before you deleted the entry.

19.6 Managing Attribute Uniqueness Constraint Entries by Using the Command Line

You can manage an attribute uniqueness constraint policy by using the command line.

The following sections explain this further:

- Creating Attribute Uniqueness Across a Directory by Using Command-Line
- Specifying Uniqueness Constraint for an Attribute by Using Command-Line
- Creating Attribute Uniqueness Across One Subtree by Using Command-Line
- Creating Attribute Uniqueness Across One Object Class by Using Command-Line
- Modifying Attribute Uniqueness Constraint Entries by Using Command-Line
- Deleting Attribute Uniqueness Constraint Entries by Using Command-Line
- Enabling and Disabling Attribute Uniqueness by Using Command-Line

19.6.1 Creating Attribute Uniqueness Across a Directory by Using Command-Line

To create an instance of attribute uniqueness across an entire directory, specify an attribute name for which you want to enforce value uniqueness.

For example, to make employee identifiers unique for all US employees at MyCompany, you would follow these steps:

1. Create an attribute uniqueness constraint entry (in LDIF format) as follows:

```
dn: cn=constraint1, cn=unique, cn=common, cn=products, cn=oraclecontext
objectclass: orclUniqueConfig
orcluniqueattrname: employeenumber
```

```
orcluniquesubtree: o=MyCompany, c=US
orcluniqueobjectclass: person
```

2. Apply the attribute uniqueness feature by loading the attribute uniqueness constraint entry as follows:

   ```
   ldapadd -h host -p port -D DN -q -f constraint1.ldif
   ```

3. Restart the directory server.

19.6.2 Specifying Uniqueness Constraint for an Attribute by Using Command-Line

Using Command-Line you can specify uniqueness constraint for an attribute.

The following LDIF file, `uniquenessConstraint.ldif`, specifies a uniqueness constraint for the `orclcommonusernickname` attribute:

```
# Use this LDIF file to set up a uniqueness constraint on the nickname
# attribute within the user search base.
# Before running the script, change the following parameters in the LDIF file.
#  <userid_attribute> -  Specify the name of the attribute that holds the user
# id. This value should be the same as the orclcommonusernickname attribute
# configured for the realm.#  <dn _f_user_serach_base> - Specify the user search base in
which the
# uniqueness constraint should be enforced.
#
dn: cn=<userid_attribute> ,cn=unique,cn=common,cn=Products, cn=OracleContext
changetype: add
objectclass: orclUniqueConfig
orcluniqueattrname: <userid _ttribute>
orcluniquesubtree: <dn_of_user_search_base>
orcluniqueenable:1
```

19.6.3 Creating Attribute Uniqueness Across One Subtree by Using Command-Line

To create an instance of attribute uniqueness across one or more subtrees, specify:

- An attribute name for which you want to enforce value uniqueness
- Subtree locations under which you want the uniqueness constraint to be enforced

For example, suppose that MyCompany hosts the directories for SubscriberCompany1 and SubscriberCompany2, and you want to enforce the uniqueness of the employee identifier attribute in SubscriberCompany1 only. When you add an entry such as `uid=dlin,ou=people,o=SubscriberCompany1,dc=MyCompany, dc=com`, you must enforce uniqueness only in the `o=SubscriberCompany1,dc=MyCompany,dc=com` subtree. Do this by listing the DN of the subtree explicitly in the attribute uniqueness constraint configuration.

In this case, the LDIF file would look like this:

```
dn: cn=constraint1, cn=unique, cn=common, cn=products, cn=oraclecontext
objectclass: orclUniqueConfig
orcluniqueattrname: employeenumber
orcluniquesubtree: o=SubscriberCompany1,dc=MyCompany,    dc=com
```

19.6.4 Creating Attribute Uniqueness Across One Object Class by Using Command-Line

To create an instance of attribute uniqueness across one object class, you need to specify an attribute name for which you want to enforce value uniqueness and an object class name.

In this case, the LDIF file would look like this:

```
dn: cn=constraint1, cn=unique, cn=common, cn=products, cn=oraclecontext
objectclass: orclUniqueConfig
orcluniqueattrname: employeenumber
orcluniqueobjectclass: person
```

Use `ldapadd` to add the entry.

```
ldapadd -D "cn=orcladmin" -q -p port -D user -f file_name
```

19.6.5 Modifying Attribute Uniqueness Constraint Entries by Using Command-Line

To modify an attribute uniqueness entry, use create an LDIF file for the entry, then use `ldapmodify` to upload it into the directory.

For example, suppose there is an existing attribute uniqueness constraint entry:

```
dn: cn=constraint1, cn=unique, cn=common, cn=products, cn=oraclecontext
objectclass: orclUniqueConfig
orcluniqueattrname: employeenumber
orcluniquesubtree: o=MyCompany, c=US
orcluniqueobjectclass: person
```

To enforce the constraint against `c=US`, instead of `o=MyCompany`, you would perform these steps:

1. Create an LDIF entry to change the `orcluniquenesssubtree`:

   ```
   dn: cn=constraint1, cn=unique, cn=common, cn=products, cn=oraclecontext
   changetype: modify
   replace: orcluniquesubtree
   orcluniquesubtree: o=Oracle Corporation, c=US
   ```

2. Use `ldapmodify` to apply the change to directory server.

   ```
   ldapmodify -D "cn=orcladmin" -q -p port -D user -f file_name
   ```

3. Restart the directory server to effect this change.

19.6.6 Deleting Attribute Uniqueness Constraint Entries by Using Command-Line

Use the `ldapdelete` command-line tool to delete an attribute uniqueness constraint policy.

1. Remove the attribute uniqueness constraint entry from the directory by using ldapdelete.

   ```
   ldapdelete -D "cn=orcladmin" -q -p port -D bind_DN \
    "cn=constraint1,cn=unique,cn=common,cn=products,cn=oraclecontext"
   ```

2. Restart the directory server to effect this change.

19.6.7 Enabling and Disabling Attribute Uniqueness by Using Command-Line

You can enable or disable attribute uniqueness for an existing attribute uniqueness constraint entry.

To enable attribute uniqueness for an existing attribute uniqueness constraint entry:

1. Set the `orcluniqueenable` attribute to `1` by using `ldapmodify`.
2. Restart the directory server to enable the policy.

To disable attribute uniqueness:

1. Set the `orcluniqueenable` attribute to `0` by using ldapmodify.
2. Restart the directory server to disable the policy.

20

Managing Knowledge References and Referrals

You can manage knowledge references and referrals in Oracle Internet Directory and configure smart referrals and default referrals.

- Introduction to Managing Knowledge References and Referrals
- About Referral Sets
- Configuring Smart Referrals
- Configuring Default Referrals

> **See Also:**
>
> Knowledge References and Referrals for an overview of directory entries, directory information trees, distinguished names, and relative distinguished names.

20.1 Introduction to Managing Knowledge References and Referrals

A knowledge reference, also called a referral, is represented in the directory as a particular type of entry. When you create a knowledge reference entry, you associate it with the `referral` object class the and `extensibleObject` object class. Typically, you create knowledge reference entries at the place in the DIT where you want to establish the partition.

A knowledge reference provides users with a referral containing an LDAP URL. You enter these URLs as values for the `ref` attribute. There can be multiple `ref` attributes specified for any knowledge reference entry. Similarly, there can be multiple knowledge reference entries in the DIT.

> **See Also:**
>
> Understanding the Concepts and Architecture of Oracle Internet Directory for an overview of directory partitioning and knowledge references and a description of a smart knowledge reference and a default knowledge reference

Referral caching is the process of storing referral information so that it can be easily accessed again and again. Suppose that a client queries Server A, which returns a referral to Server B. The client chases this referral and contacts Server B which performs the operation and returns the results to the client. Without referral caching, the next time the client makes the same query to Server A, the entire procedure is repeated, an unnecessary consumption of time and system resources.

However, if the referral information can be cached, then, in each subsequent query, the referral information can be obtained from cache and Server B can be contacted directly. This speeds up the operation.

Client-side referral caching enables each client to cache this referral information and use it to speed up of referral processing.

Referral entries are stored in a configuration file on the client. When a client establishes a session, it reads the referral information from this configuration file and stores them in a cache. This cache remains static, with no further updates being added during the session. From this point on, for every operation, the client looks up referral information in the cache.

The directory administrator prepares this configuration file for clients to use.

> **Note:**
>
> The configuration file is optional for clients. If a file is not present, then client operations involving referrals still behave correctly. Thus it is not mandatory for administrator to prepare this file. The advantage of using the configuration file is that it speeds up the client/server operations involving referrals.

20.2 About Referral Sets

The configuration file consists of one or more referral sets.

Each referral set consists of:

- The host name where a particular directory server is running
- One or more referral entries residing on that server

Each referral entry consists of a sequence of lines, each of which corresponds to one referral URL. The line separator is CR or LF.

```
ref_file=ref_file_content
ref_file_content=1*(referral_set)
referral_set=hostname       SEP      ref_entry_set   SEP
ref_entry_set=ref_entry     *(SEP    ref_entry)
ref_entry=1*(referralurl    SEP)
SEP=CR LF / LF
CR=0x0D
LF=0x0A
```

For example, consider two referral entries in a directory server running on host serverX:

```
dn: dc=example, dc=com
ref: ldap://serverA:3060/dc=example, dc=com
ref: ldap://serverB:3060/dc=example, dc=com

dn: dc=oracle, dc=com
ref: ldap://serverC:3060/dc=oracle, dc=com
ref: ldap://serverD:3060/dc=oracle, dc=com
```

Consider the following referral entry in a directory server running on host serverY:-

```
dn: dc=fiction, dc=com
ref: ldap://serverE:3060/dc=fiction, dc=com
```

The corresponding `referral.ora` file looks like this:

```
ServerX
ldap://serverA:3060/dc=example, dc=com
ldap://serverB:3060/dc=example, dc=com

ldap://serverC:3060/dc=oracle, dc=com
ldap://serverD:3060/dc=oracle, dc=com

ServerY
ldap://serverE:3060/dc=fiction, dc=com
```

20.3 Configuring Smart Referrals

A search result can contain regular entries along with knowledge references. When a user performs a search operation, Oracle Internet Directory looks for the knowledge reference entry within the specified scope of the search. If it finds the knowledge reference, then Oracle Internet Directory returns a referral to the client.

If a user performs an add, delete, or modify operation on an entry located below the knowledge reference entry, then Oracle Internet Directory returns the referral.

For example, suppose you want to partition the DIT based on the geographical location of the directory servers. In this example, assume that:

- The `c=us` naming context is held locally on Server A and Server B in the United States.
- The `c=uk` naming context is held locally on Server C and Server D in the United Kingdom.

In this case, you would configure knowledge references between these two naming contexts as follows:

1. On Server A and B in the United States, configure a knowledge reference for the `c=uk` object on Server C and Server D:

   ```
   dn: c=uk
   c: uk
   ref: ldap://host C:3060/c=uk
   ref: ldap://host D:600/c=uk
   objectclass: top
   objectclass: referral
   objectClass: extensibleObject
   ```

2. Configure a similar knowledge reference on Server C and D in the United Kingdom for the `c=us` object on Server A and Server B:

   ```
   dn: c=us
   c: us
   ref: ldap://host A:4000/c=us
   ref: ldap://host B:5000/c=us
   objectclass: top
   objectclass: referral
   objectClass: extensibleObject
   ```

Use the command-line:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

Results:

- A client querying Server A or Server B with base `o=foo,c=uk` receives a referral.
- A client querying Server C or Server D with base `o=foo,c=us` receives a referral.

- An add operation of `o=foo,c=uk` on either Server A or Server B fails. Instead, Oracle Internet Directory returns a referral.

20.4 Configuring Default Referrals

Oracle Internet Directory uses the `namingcontext` attribute in the DSE to determine every directory naming context held locally by the server. Be sure that the `namingContext` attribute correctly reflects the naming context information.

You specify default referrals by entering a value for the `ref` attribute in the DSE entry. If the `ref` attribute is not in the DSE entry, then no default referral is returned.

When configuring a default referral, do not specify the DN in the LDAP URL.

For example, suppose that the DSE entry on Server A contains the following `namingContext` value:

```
namingcontext: c=us
```

Further, suppose that the default referral is:

```
Ref: ldap://host PQR:3060
```

Now, suppose that a user enters an operation on Server A that has a base DN in the naming context `c=canada`, for example:

```
ou=marketing,o=foo,c=canada
```

This user would receive a referral to the host PQR. This is because Server A does not hold the `c=canada` base DN, and the `namingcontext` attribute in its DSE does not hold the value `c=canada`.

> **See Also:**
>
> Knowledge References and Referrals for a conceptual discussion of knowledge references

> **See Also:**
>
> Managing Naming Contexts in Oracle Internet Directory.

21

Managing Directory Schema

This chapter explains how to manager the Oracle Internet Directory object classes and attributes in the directory schema using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities.

- Introduction to Managing Directory Schema
- Managing Directory Schema by Using Oracle Directory Services Manager
- Managing Directory Schema by Using the Command Line

21.1 Introduction to Managing Directory Schema

This section introduces you to directory schema management and related topics.

This section introduces you to the following topics:

- Understanding Directory Schema Management,
- Storage Location of Schema Information in the Directory
- Understanding Object Classes
- Understanding Attributes
- Methods to Extend the Number of Attributes Associated with Entries
- Understanding Attribute Aliases
- Object Identifier Support in LDAP Operations

> **See Also:**
>
> LDAP Schema Overview in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

21.1.1 Understanding Directory Schema Management

Oracle recommends that you understand the basic concepts of directory components before attempting to add to or modify the base schema in the directory.

A directory schema:

- Contains rules about the kinds of objects you can store in the directory
- Contains rules for how directory servers and clients treat information during operations such as a search
- Helps to maintain the integrity and quality of the data stored in the directory
- Reduces duplication of data

- Provides a predictable way for directory-enabled applications to access and modify directory objects

> **Note:**
>
> Every schema object in the Oracle Internet Directory has certain limitations. For example, some objects cannot be changed. These limitations are explained as constraints and rules in this chapter.

21.1.2 Storage Location of Schema Information in the Directory

The directory schema contains all information about how data is organized in the DIT—that is, metadata such as that for an object class, an attribute, a matching rule, and syntax. This information is stored in a special class of entry called a subentry. More specifically, Oracle Internet Directory, following LDAP Version 3 standards, stores this information in subentries of type `subSchemaSubentry`.

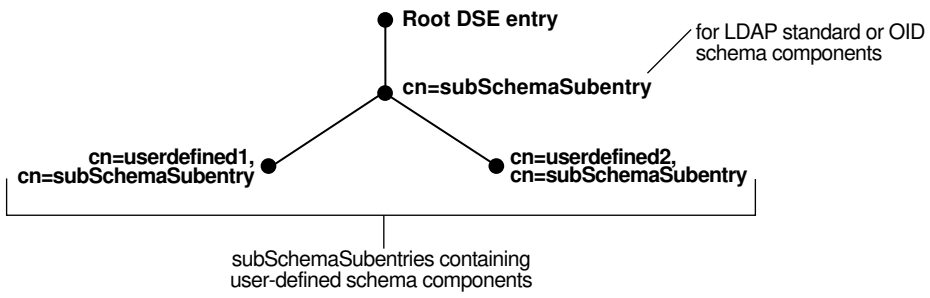
You can add new object classes and attribute types to a subentry of type `subSchemaSubentry`. You cannot add new matching rules and syntaxes beyond those already supported by Oracle Internet Directory.

Before 10g (10.1.4.0.1) there was only one `subSchemaSubentry`, directly under the root DSE entry, called `cn=subSchemaSubentry`, and you always added object classes and attribute types directly to it. The entry `cn=subSchemaSubentry` can now have subordinate entries. Any application using Oracle Internet Directory that must add schema components can create its own `subSchemaSubentry` under `cn=subSchemaSubentry` and add the schema components to it. Figure 21-1 shows some `subSchemaSubentry` entries that were defined by applications, such as `cn=userdefined1,cn=subschemasubentry` and `cn=userdefined2,cn=subschemasubentry`.

Best practice is to create one subordinate entry under `subschemasubentry` for one application or a group of applications. All the attributeTypes and objectClasses related to the application should be put under its corresponding entry under `cn=subschemasubentry`.

All the schema entries are listed in the root DSE attribute `subschemasubentry`.

Figure 21-1 Location of Schema Components in Entries of Type subSchemaSubentry



You cannot use `bulkload` to add `subSchemaSubentry` entries. You must use `ldapadd`.

21.1.3 Understanding Object Classes

This section introduces you to object classes and related topics.

This section introduces you to the following topics:

- About Object Classes,
- Add Object Classes,
- Types of Modification to an Existing Object Class,
- Limitations for Deleting Object Classes,

21.1.3.1 About Object Classes

When you add an entry, you associate it with one or more object classes. Each object class contains attributes that you want to associate with the new entry.

For example, if you are creating an entry for an employee, you can associate it with the `person` object class. This object class contains many of the attributes that you want to associate with that employee entry, including, for example, name, address, and telephone number.

> **Note:**
>
> The attribute page in the Schema tab in Oracle Directory Services Manager includes a **Referenced By** section that lists the object classes that directly reference an attribute

Each object class derives from a hierarchy of superclasses, and it inherits attributes from these superclasses. By default, all object classes inherit from the `top` object class. When you assign an object class to an entry, the entry inherits all of the attributes of both that object class and its superclasses.

The attributes that entries inherit from a super class may be either mandatory or optional. Values for optional attributes need not be present in the directory entry.

You can specify for any object class whether an attribute is mandatory or optional; however, the characteristic you specify is binding only for that object class. If you place the attribute in another object class, you can again specify whether the attribute is mandatory or optional for that object class. You can:

- Add a new, nonstandard object class and assign it existing attributes
- Select from existing standard object classes
- Modify an existing object class, assigning it a different set of attributes
- Add and modify existing attributes

> **See Also:**
>
> LDAP Schema Overview in *Oracle® Fusion Middleware Reference for Oracle Identity Management* for a list of schema elements installed with Oracle Internet Directory

Entries must be added in a top-down sequence—that is, when you add an entry, all of its parent entries must already exist in the directory. Similarly, when you add entries that reference object classes and attributes, those referenced object classes and attributes must already exist

in the directory schema. In most cases this is not a problem because the directory server is delivered with a full set of standard directory objects.

When you add or perform an operation on an entry, you do not need to specify the entire hierarchy of superclasses associated with that entry. You can specify only the leaf object classes. Oracle Internet Directory resolves the hierarchy for the leaf object classes and enforces the information model constraints. This is often called **object class explosion**. For example, the `inetOrgPerson` object class has `top`, `person` and `organizationalPerson` as its superclasses. When you create an entry for a person, you must specify only `inetOrgPerson` as the object class. Oracle Internet Directory then enforces the schema constraints defined by the respective superclasses, namely, `top`, `person`, and `organizationalPerson`.

21.1.3.2 Add Object Classes

When you add object classes, keep the following in mind:

- Every structural object class must have `top` as a superclass.
- The name and the object identifier of an object class must be unique across all the schema components. The actual attribute name and any attribute name aliases must be unique across all attribute names and attribute aliases. The Object Identifier must begin with the unique identifier 2.16.840.1.113894 followed by either the Oracle-supplied prefix.9999 or a site-specific prefix.
- Schema components referred to in the object class, such as superclasses, must already exist.
- The superclass of an abstract object class must be abstract also.
- It is possible to redefine mandatory attributes in a superclass into optional attributes in the new object class. Conversely, optional attributes in a superclass can be redefined into mandatory attributes in the new object class.

> **See Also:**
>
> Subclasses, Superclasses, and Inheritance for a conceptual discussion of these terms

21.1.3.3 Types of Modification to an Existing Object Class

You can perform modifications to an existing object class through Oracle Directory Services Manager and through the command-line tools.

You can make these changes to an object class in the following ways:

- Change a mandatory attribute into an optional attribute
- Add optional attributes
- Add additional superclasses
- Convert *abstract* object classes into *structural* or *auxiliary* object classes unless the abstract object class is a superclass to another abstract object class

When you modify object classes, keep these guidelines in mind:

- You cannot modify an object class that is part of the standard LDAP schema. You can, however, modify user-defined object classes.

- If existing object classes do not have the attributes you need, you can create an auxiliary object class and associate the needed attributes with that object class.
- You cannot add additional mandatory attributes to an existing object class.
- You should not modify object classes in the base schema.
- You cannot remove attributes or superclasses from an existing object class.
- You cannot convert structural object classes to other object class types.
- You should not modify an object class if there are entries already associated with it.

21.1.3.4 Limitations for Deleting Object Classes

The limitations on deleting object classes are:

- You cannot delete object classes from the base schema.
- You can delete object classes that are not in the base schema if they are not directly or indirectly referenced by other schema components. For example, there may be some directory entries referring to these object classes. Deleting these object classes renders these entries inaccessible.

21.1.4 Understanding Attributes

You can find introduction on how to add, modify or delete attributes.

This section introduces you to the following topics:

- Introduction to Attributes,
- Rules for Adding Attributes,
- Rules for Modifying Attributes,
- Rules for Deleting Attributes,
- Index option in Oracle Internet Directory to Search Attributes ,

21.1.4.1 Introduction to Attributes

You must understand attributes from a conceptual standpoint before attempting operations involving attributes.

In most cases, the attributes available in the base schema will suit the needs of your organization. However, if you decide to use an attribute not in the base schema, you can add a new attribute or modify an existing one.

By default, attributes are multivalued. You can specify an attribute as single-valued by using either Oracle Directory Services Manager or command-line tools.

> **Note:**
>
> The maximum length of a non-binary attribute value is 4000 bytes.

See Also:

- Understanding the Concept of Attributes in Oracle Internet Directory for a conceptual discussion of attributes
- Attribute Options for information about attribute options
- Attribute Syntax in *Oracle Fusion Middleware Reference for Oracle Identity Management* for information about using syntax to specify the size of the attribute value

.

21.1.4.2 Rules for Adding Attributes

The rules for adding attributes are:

- The name and the object identifier of an attribute must be unique across all the schema components.
- Syntax and matching rules must agree.
- Any super attributes must already exist.
- The length of an attribute name must not exceed 127 characters.

21.1.4.3 Rules for Modifying Attributes

The rules for modifying attributes are:

- The name and the object identifier of an attribute must be unique across all the schema components.
- The syntax of an attribute cannot be modified.
- A single-valued attribute can be made multi-valued, but a multi-valued attribute cannot be made single-valued.
- You cannot modify or delete base schema attributes.

21.1.4.4 Rules for Deleting Attributes

The rules for deleting attributes are:

- You can delete only user-defined attributes. Do not delete attributes from the base schema.
- You can delete any attribute that is not referenced directly or indirectly by some other schema component.

  If you delete an attribute that is referenced by any entry, that entry will no longer be available for directory operations.

> **Note:**
>
> The attribute page in the Schema tab in Oracle Directory Services Manager includes a **Referenced By** section that lists the object classes that directly reference an attribute

> **See Also:**
>
> Attribute Syntax in *Oracle Fusion Middleware Reference for Oracle Identity Management* for information about using syntax to specify the size of the attribute value.

21.1.4.5 Index option in Oracle Internet Directory to Search Attributes

Oracle Internet Directory uses indexes to make attributes available for searches. When Oracle Internet Directory is installed, the entry `cn=catalogs` lists available attributes that can be used in a search.

You can use the autocatalog feature, which is enabled by default in fresh installs. You can also enable it if you have upgraded from a previous release. When this feature is enabled, Oracle Internet Directory automatically indexes attributes when you search for them. You can disable the feature by setting the `orclautocatalog` attribute of the DSA configuration entry to 0. To modify this attribute from the command line, see Setting System Configuration Attributes by Using ldapmodify . To modify this attribute by using Oracle Enterprise Manager Fusion Middleware Control, see Configuring Shared Properties.

If the autocatalog feature is not enabled, and you want to use additional attributes in search filters, you must add them to the catalog entry.

You can index an attribute or drop an index from an attribute by using either `ldapmodify` or `catalog`. You can also manage attribute indexes by using Oracle Directory Services Manager.

Note:

- When autocatalog is disabled, if you attempt to perform a search with a non-indexed attribute specified as a required attribute, the server returns a `Function not implemented` or `DSA unwilling to perform` error.
- Autocatalog is performed only when a valid user connects to Oracle Internet Directory. Anonymous bind search does not trigger the autocatalog feature. Also, this event is audited.When two or more clients search the same non-cataloged attribute at the same time, only one thread initiates the catalog process. The other threads return LDAP error 53 and the additional information: `OID-5018: Cataloging for attribute is already in progress`. The cataloging thread waits for the cataloging (separate) process to finish or until configured `orclmaxserverresptime`. When `orclmaxserverresptime` has elapsed, the cataloging thread returns LDAP error 53 and the additional info: "`OID-5018: Cataloging for attribute is already in progress`.The catalog of each attribute is created once only on every directory so this is a one time effect for every non-cataloged attribute. If a search is interrupted due to a client side time-out or an explicit kill command, the in-progress catalog will continue and the subsequent searches on same attribute will succeed once the catalog processing completes.
- You can use Oracle Directory Services Manager to index an attribute only if it has not been used yet. You cannot use Oracle Directory Services Manager to index an attribute that is already in use. To index an attribute that is in use, use ldapmodify as described in Indexing an Attribute by Using ldapmodify, or the Catalog Management tool, as described in Indexing an Attribute by Using the Catalog Management Tool.

Note:

You can index only those attributes that have:

- An equality matching rule
- Matching rules supported by Oracle Internet Directory as listed in Attribute Matching Rules in *Oracle Fusion Middleware Reference for Oracle Identity Management*.
- Less than 128 characters in their names

21.1.5 Methods to Extend the Number of Attributes Associated with Entries

You can understand how to extend the number of attributes associated with entries.

This section contains these topics:

- Extending the Number of Attributes Associated with Entries,
- Extending the Number of Attributes before Creating Entries in the Directory
- Specifications to Extend the Number of Attributes for Existing Entries by Creating an Auxiliary Object Class

- Specifications to Extend the Number of Attributes for Existing Entries by Creating a Content Rule
- Rules for Creating and Modifying Content Rules,
- Schema Enforcement When Using Content Rules,
- Searches for Object Classes Listed in Content Rules,

21.1.5.1 Extending the Number of Attributes Associated with Entries

You can extend the number of attributes for entries. The method you use depends on whether the entries already exist.

If the entry does not yet exist, you can extend the number of attributes before creating the entry in the directory.

For an existing entry, there are two ways to extend the attributes associated with it.

- Add names of object classes to the list in the `objectclass` attribute for each entry. If your directory is relatively small, then this can be a desirable method because it enables searches for entries based on that attribute. However, if your directory is large, then entering the names of object classes to the `objectclass` attribute can be very painstaking.
- Use content rules. This may be a more efficient way to extend the content of entries in a large directory.

21.1.5.2 Extending the Number of Attributes before Creating Entries in the Directory

At installation, Oracle Internet Directory provides standard LDAP object classes and several proprietary object classes. You cannot add mandatory attributes to the sets of attributes belonging to these predefined object classes. If a given object class does not contain all the attributes that you want for an entry, then you can do one of the following:

- Define a new (base) object class
- Define an object subclass

> **See Also:**
>
> – Oracle Identity Management LDAP Object Classes in *Oracle Fusion Middleware Reference for Oracle Identity Management* for a list of object classes in the schema installed with Oracle Internet Directory.
>
> – Adding Object Classes by Using Oracle Directory Services Manager for instructions on how to define a new object class or object subclass

21.1.5.3 Specifications to Extend the Number of Attributes for Existing Entries by Creating an Auxiliary Object Class

You can create an auxiliary object class containing the additional attributes you want for your entry, and then associate that auxiliary object class with the entry. You associate the auxiliary object class with the entry by specifying it in the `objectclass` attribute for the entry.

> **See Also:**
>
> - Adding Object Classes by Using Oracle Directory Services Manager for instructions on creating auxiliary object classes
> - Managing Directory Entries in Oracle Internet Directory for instructions on associating an object class with an entry

21.1.5.4 Specifications to Extend the Number of Attributes for Existing Entries by Creating a Content Rule

A content rule, following your specifications, determines the kind of content allowed in any entry that is associated with a particular structural object class. For example, you can specify that any entry associated with the `person` object class must have, in addition to the attributes in that object class, other attributes as well. The additional attributes can be those of an auxiliary object class, and they can be either mandatory or optional.

Whereas you must list auxiliary classes in the entry—which can be an administrative burden—you do not need to list content rules in the entry.

In addition to the structural object class to which it applies, a content rule can also indicate:

- Auxiliary object classes allowed for entries governed by the rule
- Mandatory attributes, in addition to those called for by the structural and auxiliary object classes, required for entries governed by the DIT content rule
- Optional attributes permitted for entries governed by the DIT content rule, in addition to those called for by structural and auxiliary object classes

This section tells you how to manage content rules by using Oracle Directory Services Manager and command-line tools.

During the process of defining a new content rule, the directory server validates the syntax and ensures that the attributes and object classes listed in the content rule have been defined in the directory.

Content rules can be specified for structural object classes only. The name of the object class is case-insensitive.

You can specify more than one content rule for each structural object class provided the content rules have different labels associated with them.

To modify an existing definition of a content rule, the client must first delete the existing definition and then add the new definition. Simple replacement of a content rule by using the `replace` command is not allowed.

To delete a content rule, the client must specify only the structural object class and the alphanumeric object identifier of the content rule. Optionally, the client can also specify the associated version of the content rule to be deleted.

21.1.5.5 Rules for Creating and Modifying Content Rules

Content rules are defined as values of the `DITContentRule` attribute in the subschema subentry (`cn=subschemasubentry`). They must conform to these rules:

- The structural object class of the entry identifies the content rule applicable for the entry. If no content rule is present for a structural object class, then entries associated with that object class contain only the attributes permitted by the structural object class definition.
- Because a content rule is associated with a structural object class, all entries of the same structural object class have the same content rule regardless of their location in the DIT.
- The content of an entry must be consistent with the object classes listed in the `objectClass` attribute of that entry. More specifically:
  - Mandatory attributes of object classes listed in the `objectClass` attribute must always be present in the entry.
  - Optional attributes of auxiliary object classes indicated by the content rule can also be present even if the `objectClass` attribute does not list these auxiliary object classes.

> **See Also:**
>
> Specifications to Extend the Number of Attributes for Existing Entries by Creating an Auxiliary Object Class for instructions on creating and managing content rules

21.1.5.6 Schema Enforcement When Using Content Rules

When validating an object for schema consistency, the directory server uses the content rule for the structural object class of the entry. It also uses all the other object classes listed in the entry.

If more than one content rule exists for an object class, then, when adding or modifying an entry, or when bulkloading data, the following rules apply.

- An entry can have attributes from all the auxiliary object classes listed in the various content rules. Not specifying an object class in the content rule does not restrict a client from explicitly adding an auxiliary object class in directory entries.
- An entry must contain values for all the mandatory attributes listed in:
  - The content rules
  - The object classes associated with the entry
  - The auxiliary object classes listed in the content rule applicable to the entry
- Optionally, an entry can contain values for any or all the optional attributes listed in:
  - The content rule
  - The object classes listed in the entry
  - The auxiliary object classes listed in the content rule applicable for the entry
- If any attribute is specified as mandatory, then it overrides any other definition that defines it as optional.

21.1.5.7 Searches for Object Classes Listed in Content Rules

Because the auxiliary object classes listed in content rules are not listed in the `objectclass` attribute for an entry, you cannot list those object classes as filters when you search for entries. Instead, base your searches on the structural object class that you are interested in. If you

must base your search on an auxiliary object class, then add that auxiliary object class to the `objectclass` attribute in the user objects explicitly.

For example, a content rule for structural object class `inetOrgPerson` may specify an auxiliary object class `orclUser`. However, this does not mean that every `inetOrgPerson` entry in the directory contains `orclUser` as a value of the `objectclass` attribute. As a result, the search with the filter `objectclass=orclUser` fails. Instead of querying for an auxiliary object class contained in the content rule, you should query for structural object classes—for example, `objectclass=inetOrgPerson`.

To base a search on `objectclass=orcluser`, add `orclUser` as one of the values of `objectclass` attribute in each entry.

These considerations apply also to filters used in access control policies. If you are using a content rule to associate additional auxiliary object classes, then use only the structural object classes in the search filters.

21.1.6 Understanding Attribute Aliases

You can create aliases for attribute names. For example. you could create the user-friendly alias `surname` for the attribute `sn`. After you create an alias for an attribute name, a user can specify the alias instead of the attribute name in an LDAP operation.

You define an alias for an attribute in the LDAP schema definition of the attribute. The directory schema operational attribute `attributeTypes` has been enhanced to allow you to include aliases in the attribute name list. In previous releases, the format for an attribute name list was:

```
attributeTypes=( ObjectIdentifier NAME 'AttributeName'  ... )
```

You may optionally specify:

```
attributeTypes=( ObjectIdentifier NAME ( 'AttributeName' 'Alias1' 'Alias2' ...) ... )
```

This is consistent with the LDAP protocol as specified by RFC 2251 and RFC 2252. In the attribute name list, the first item is recognized as the name of the attribute and rest of the items in the list are recognized as attribute aliases. For example, to specify the alias `surname` for the attribute `sn`, you would change the schema definition for `sn` from:

```
attributeTypes=( 2.5.4.4 NAME 'sn' SUP name )
```

to:

```
attributeTypes=( 2.5.4.4 NAME ( 'sn' 'surname') SUP name )
```

The following rules apply to attribute aliases:

- An attribute alias name must be unique throughout all the actual attribute names and other attribute aliases across all the schema components.When you define an attribute, the first value in the `attributeTypes` definition `NAME` field must be the actual attribute name. You define attribute aliases in the `NAME` field after the actual attribute name.
- Attribute alias names follow the same syntax rules as attribute names.
- You delete an attribute alias by redefining the attribute without the alias.

See Managing Directory Schema by Using the Command Line.

> **See Also:**
>
> Managing Alias Entries for information about alias entries.

21.1.7 Object Identifier Support in LDAP Operations

You can find how to substitute object identifiers for attribute names.

Users can substitute object identifiers for attribute names in the same way as attribute aliases.

21.2 Managing Directory Schema by Using Oracle Directory Services Manager

You can perform various operations on object classes like searching, adding, deleting using Oracle Directory Services Manager.

This section contains the following topics:

- Searching for Object Classes by Using Oracle Directory Services Manager
- Adding Object Classes by Using Oracle Directory Services Manager
- Deleting Object Classes by Using Oracle Directory Services Manager
- Viewing Properties of Object Classes by Using Oracle Directory Services Manager
- Adding a New Attribute by Using Oracle Directory Services Manager
- Modifying an Attribute by Using Oracle Directory Services Manager
- Deleting an Attribute by Using Oracle Directory Services Manager
- Viewing All Directory Attributes by Using Oracle Directory Services Manager
- Searching for Attributes by Using Oracle Directory Services Manager
- Adding an Index to a New Attribute by Using Oracle Directory Services Manager
- Adding an Index to an Existing Attribute by Using Oracle Directory Services Manager
- Adding an Index to an Existing Attribute by Using Oracle Directory Services Manager
- Creating a Content Rule by Using Oracle Directory Services Manager
- Modifying a Content Rule by Using Oracle Directory Services Manager
- Viewing Matching Rules by Using Oracle Directory Services Manager
- Viewing Syntaxes by Using Oracle Directory Services Manager

21.2.1 Searching for Object Classes by Using Oracle Directory Services Manager

To search for object classes by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.

3. Expand the Object Classes panel on the left.
4. Enter a keyword in the **Search** field and click **Go**. The list of object classes matching the keyword is displayed in the left panel.

   You can use "`*`" and "`?`" as wildcards. For example, if you enter `*person`, the search returns all the object classes ending with `person`.

   Select **Clear search text** to dismiss the search and return to the complete list of object classes.

21.2.2 Adding Object Classes by Using Oracle Directory Services Manager

To add object classes by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. Expand the Object Classes panel on the left and, in the toolbar, choose **Create**. The New Object Class dialog box appears.

   Alternatively, in the **Object Classes** panel, select an object class that is similar to one you would like to create, and then choose **Create Like**. The New Object Class dialog box displays the attributes of the selected object class. You can create the new object class by using this one as a template.
4. In the New Object Class dialog box, enter the information in the fields.
5. Choose **Create**.

> **See Also:**
>
> - Understanding the Types of Object Class
> - Subclasses, Superclasses, and Inheritance
> - Oracle Directory Services Manager online help for further details about adding object classes

21.2.3 Modifying Object Classes by Using Oracle Directory Services Manager

You can add optional, but not mandatory, attributes to an object class already in use by entries. If you add optional attributes to an object class already in use, then no special rules apply, and they are added as empty attributes to those entries.

To modify an object class:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. Expand the Object Classes panel on the left. Use the scroll bar to move through the alphabetical list of object classes. You can also search for an object class as described in Searching for Object Classes by Using Oracle Directory Services Manager.

4. Click the object class you want to modify. The Object Class tab appears on the right side of the page.
5. To add or delete a superclass or attribute, select it in the **Superclass**, **Mandatory Attributes**, or **Optional Attributes** list and choose **Add** or **Delete** in the toolbar above the list

   Choose **Make Optional** in the **Mandatory Attributes** toolbar to make the attribute optional.

   To edit a superclass or attribute, select it in the **Superclass**, **Mandatory Attributes**, or **Optional Attributes** list and choose **Edit**. An Object Class or Attribute dialog appears. Use it to make changes to the superclass or attribute. In the Object Class dialog, you can add, delete, or edit the superclass, mandatory attributes, or optional attributes of the superclass. Click **OK** in each dialog after making change.
6. Choose **Apply** in the Object Class page to apply changes, or **Revert** to abandon changes.

> **See Also:**
>
> - Understanding Object Classes in Oracle Internet Directory
> - Subclasses, Superclasses, and Inheritance

> **Note:**
>
> You can add attributes to an auxiliary object class or a user-defined structural object class.
>
> **See Also:** Adding a New Attribute to an Auxiliary or User-Defined Object Class by Using Command-Line Tools for an example of adding attributes to an auxiliary object class

21.2.4 Deleting Object Classes by Using Oracle Directory Services Manager

> **Note:**
>
> Oracle recommends that you do not delete object classes from the base schema. If you delete an object class that is referenced by any entries, those entries then become inaccessible.
>
> Deleting object classes from the base schema might also cause Oracle Directory Services Manager to malfunction.

To delete an object class by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.

3. Click **+** next to **Object Classes** to expand the Object Classes panel. Use the scroll bar to move through the alphabetical list of Object Classes.
4. Select the object class you want to delete.
5. Choose **Delete** from the toolbar, then click **Delete** in the confirmation dialog.

21.2.5 Viewing Properties of Object Classes by Using Oracle Directory Services Manager

To view all object classes in the schema:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. Click **+** next to **Object Classes** to expand the Object Classes panel. Use the scroll bar to move through the alphabetical list of Object Classes.
4. Click the Object Class you want to view. The Object Class tab appears on the right side of the page.
5. To see more detail about a superclass or attribute of the object class, select the item and click Edit. Click Cancel to return to the object class panel

21.2.6 Adding a New Attribute by Using Oracle Directory Services Manager

To add a new attribute:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. If necessary, expand the **Attributes** pane, on the left, then choose the **Create** button in the toolbar. The New Attribute Type dialog box appears.

   Alternatively, in the **Attributes** panel, select an attribute that is similar to one you would like to create, and then choose **Create Like**. The New Attribute Type dialog box displays the attributes of the selected attribute. You can create the new attribute by using this one as a template.

   > **Tip:**
   >
   > Because equality, syntax, and matching rules are numerous and complex, it may be simpler to copy these characteristics from a similar existing attribute.

4. Enter values in each of the fields.
5. Choose **Apply**.

> **Note:**
>
> To use this attribute, remember to declare it to be part of the attribute set for an object class. You do this in the Object Classes pane. For further instructions, see Modifying Object Classes by Using Oracle Directory Services Manager and Modifying Object Classes by Using Command-Line Tools.

21.2.7 Modifying an Attribute by Using Oracle Directory Services Manager

To modify an attribute by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. In the **Attributes** panel, select an attribute that you would like to modify. The attribute page appears on the right.
4. Modify or add information in editable fields, if any, in the attribute tab.
5. Choose **Apply**.

21.2.8 Deleting an Attribute by Using Oracle Directory Services Manager

> **Note:**
>
> Oracle recommends that you not delete attributes or object classes from the base schema.
>
> Deleting attributes or object classes from the base schema might cause Oracle Directory Services Manager to malfunction.

To delete an attribute:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. In the **Attributes** panel, select an attribute that you would like to delete.
4. Choose **Delete** from the toolbar in the left panel, then click **Delete** in the confirmation dialog.

21.2.9 Viewing All Directory Attributes by Using Oracle Directory Services Manager

To view attributes by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.

3. Expand **Attributes**.
4. Use the scrollbar in the **Attributes** panel to move through the alphabetical list.

> **See Also:**
>
> Viewing Properties of Object Classes by Using Oracle Directory Services Manager for instructions about how to view attributes for a specific object class.

21.2.10 Searching for Attributes by Using Oracle Directory Services Manager

To search for attributes by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Schema**.
3. Expand the **Attributes** list in the left pane.
4. Enter a search term in the Search field in the left pane. You can use an asterisk (`*`) or question mark (`?`) as a wildcard. Click the **Go** icon or press `Enter` on your keyboard.
5. The results of your search appear in the list in the left pane. To dismiss the search and return to the complete list of attributes, click the **Clear search text** icon.

21.2.11 Adding an Index to a New Attribute by Using Oracle Directory Services Manager

To add an index to an attribute:

1. Create an attribute as described in Adding a New Attribute by Using Oracle Directory Services Manager or in Adding and Modifying Attributes by Using ldapmodify.
2. In the New Attribute Type dialog box, select the **Indexed** box.

21.2.12 Adding an Index to an Existing Attribute by Using Oracle Directory Services Manager

You can use Oracle Directory Services Manager to add an index to an existing attribute only if that attribute is not in use yet.

To add an index to an existing attribute:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Schema**.
3. Select an attribute that is not currently indexed that you want to add the index to.
4. Click the **The attribute will be cataloged/decataloged** icon. When the Confirm Dialog appears, click **Confirm**.
5. The Indexed box indicates that the attribute is indexed.

21.2.13 Dropping an Index from an Attribute by Using Oracle Directory Services Manager

To drop an index from an attribute:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Schema**.
3. Select an attribute that is currently indexed that you want to drop the index from.
4. Click the **The attribute will be cataloged/decataloged** icon. When the Confirm Dialog appears, click **Confirm**.
5. The Indexed box indicates that the attribute is no longer indexed.

21.2.14 Creating a Content Rule by Using Oracle Directory Services Manager

To create a content rule:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. In the left pane, expand the **Content Rules** list.
4. Choose **Create**. The New Content Rule dialog box appears.
5. In the New Content Rule dialog box, add information in the **Structural Object Class** and **Object ID** fields. Optionally, add a label.

   Choose the **Add**, **Delete**, or **Edit** icons in the toolbar above the **Auxiliary Classes**, **Mandatory Attributes**, or **Optional Attributes** list to add or delete an item. When adding, select the auxiliary class, mandatory attribute or optional attribute from the dialog. Use the search function if necessary.
6. Alternatively, in the **Content Rules** panel, select a content rule that is similar to one you would like to create, and then choose **Create Like**. The New Content Rule dialog box displays the attributes of the selected content rule. You can create the new content rule by using this one as a template.
7. Choose **OK** to add the content rule.

21.2.15 Modifying a Content Rule by Using Oracle Directory Services Manager

To modify a content rule:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. In the left pane, expand the **Content Rules** list.

4. Select the content rule you want to modify. You can search for a content rule by entering a keyword in the search field, in the same way you search for object classes. See Searching for Object Classes by Using Oracle Directory Services Manager.
5. Modify values in the appropriate fields in the content rule tab.

   Choose the **Add**, **Delete**, or **Edit** icons in the toolbar above the **Auxiliary Classes**, **Mandatory Attributes**, or **Optional Attributes** list to add or delete an item. When adding, select the auxiliary class, mandatory attribute or optional attribute from the dialog. Use the search function if necessary
6. Choose **Apply** to make the changes effective or choose **Revert** to abandon the changes.

21.2.16 Viewing Matching Rules by Using Oracle Directory Services Manager

> **Note:**
>
> Matching rules cannot be modified.

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. Expand the **Matching Rules** list. Matching rules are shown in the list.
4. You can search for a matching rule by entering a keyword in the search field, in the same way you search for object classes. See Searching for Object Classes by Using Oracle Directory Services Manager.
5. Select a matching rule to see its details in the matching rule tab page on the right.

21.2.17 Viewing Syntaxes by Using Oracle Directory Services Manager

> **Note:**
>
> Syntaxes cannot be modified.

To view syntaxes by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Schema**.
3. Expand the **Syntaxes** list. Syntax names are shown in the list.
4. Select a syntax to see its details in the syntax tab page on the right.

21.3 Managing Directory Schema by Using the Command Line

Using command line utilities, you can view schema, add new object class, add new attributes to an auxilary object class and so on.

This section contains the following topics:

- Viewing the Schema by Using ldapsearch
- Adding a New Object Class by Using Command-Line Tools
- Adding a New Attribute to an Auxiliary or User-Defined Object Class by Using Command-Line Tools
- Modifying Object Classes by Using Command-Line Tools
- Adding and Modifying Attributes by Using ldapmodify
- Deleting Attributes by Using ldapmodify
- Indexing an Attribute by Using ldapmodify
- Dropping an Index from an Attribute by Using ldapmodify
- Indexing an Attribute by Using the Catalog Management Tool
- Adding a New Attribute With Attribute Aliases by Using the Command Line
- Adding or Modifying Attribute Aliases in Existing Attributes by Using the Command Line
- Deleting Attribute Aliases by Using the Command Line
- Using Attribute Aliases with LDAP Commands
- Managing Content Rules by Using Command-Line Tools
- Viewing Matching Rules by Using ldapsearch
- Viewing Syntaxes by Using ldapsearch

21.3.1 Viewing the Schema by Using ldapsearch

You can write the schema to a file by typing:

```
ldapsearch -h OID_host -p OID_port -q -L -D "cn=orcladmin" \
     -b "cn=subschemasubentry" -s base "objectclass=*" > schema.ldif
```

21.3.2 Adding a New Object Class by Using Command-Line Tools

In this example, an LDIF input file, `new_object_class.ldif`, contains data similar to this:

```
dn: cn=subschemasubentry
changetype: modify
add: objectclasses
objectclasses: ( 2.16.840.1.113894.9999.12345 NAME 'myobjclass' SUP top STRUCTURAL MUST
( cn $ sn )
 MAY ( telephonenumber $ givenname $ myattr ) )
```

Be sure to leave the mandatory space between the opening and closing parentheses and the object identifier.

To load the file, enter this command:

```
ldapmodify -D "cn=orcladmin" -q -h myhost -p 3060 -f new_object_class.ldif
```

This example:

- Adds the *structural* object class named `myobjclass`
- Gives it an object identifier of `2.16.840.1.113894.9999.12345`.
- Specifies `top` as its superclass
- Specifies `cn` and `sn` as mandatory attributes
- Allows `telephonenumber`, `givenname`, and `myattr` as optional attributes

Note that all the attributes mentioned must exist before the execution of the command.

To create an *abstract* object class, follow the previous example, replacing the word `STRUCTURAL` with the word `ABSTRACT`.

21.3.3 Adding a New Attribute to an Auxiliary or User-Defined Object Class by Using Command-Line Tools

To add a new attribute to either an auxiliary object class or a user-defined structural object class, use `ldapmodify`. This example deletes the old object class definition and adds the new definition in a compound modify operation. The change is committed by the directory server in one transaction. Existing data is not affected. The input file should be as follows:

```
dn: cn=subschemasubentry
changetype: modify
delete: objectclasses
objectclasses: old value
-
add: objectclasses
objectclasses: new value
```

For example, to add the attribute `changes` to the existing object class `country`, the input file would be:

```
dn: cn=subschemasubentry
changetype: modify
delete: objectclasses
objectclasses:
 ( 2.16.840.1.113894.9999.12345 NAME 'country' SUP top STRUCTURAL MUST c MAY
 ( searchGuide $ description  )  )
-
add: objectclasses
objectclasses:
 ( 2.16.840.1.113894.9999.12345 NAME 'country' SUP top STRUCTURAL MUST c MAY
 ( searchGuide $ description  $ changes )  )
```

To load the file, enter this command:

```
ldapmodify -D "cn=orcladmin" -q -h myhost -p 3060 -f new_attribute.ldif
```

21.3.4 Modifying Object Classes by Using Command-Line Tools

To add or modify schema components, use `ldapmodify`.

> **See Also:**
>
> The ldapmodify command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

21.3.5 Adding and Modifying Attributes by Using ldapmodify

To add a new attribute to the schema by using `ldapmodify`, type a command similar to the following at the system prompt:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

The LDIF file contains data similar to this:

```
dn: cn=subschemasubentry
changetype: modify
add: attributetypes
attributetypes: ( 1.2.3.4.5 NAME 'myattr' SYNTAX
                 '1.3.6.1.4.1.1466.115.121.1.38' )
```

To specify an attribute as single-valued, include in the attribute definition entry in the LDIF file the keyword SINGLE-VALUE with surrounding white space.

You can find a given syntax Object ID by using either Oracle Directory Services Manager or the ldapsearch command line tool.

> **See Also:**
>
> - The `ldapmodify` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for a detailed explanation of ldapmodify and its options
> - Viewing Syntaxes by Using ldapsearch for instructions on how to view syntaxes by using either **Oracle Directory Services Manager** or `ldapsearch`

21.3.6 Deleting Attributes by Using ldapmodify

> **Note:**
>
> You can delete only user-defined attributes. Do not delete attributes from the base schema.

To delete an attribute by using ldapmodify, type a command similar to the following at the system prompt:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

The LDIF file contains data similar to this:

```
dn: cn=subschemasubentry
changetype: modify
delete: attributetypes
attributetypes: ( 1.2.3.4.5 NAME 'myattr' SYNTAX
                 '1.3.6.1.4.1.1466.115.121.1.38' )
```

You can find a given syntax Object ID by using either Oracle Directory Services Manager or the ldapsearch command line tool.

> **See Also:**
>
> - The `ldapmodify` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for a detailed explanation of ldapmodify and its options
> - Viewing Syntaxes by Using ldapsearch or Viewing Syntaxes by Using Oracle Directory Services Manager for instructions on how to view syntaxes

21.3.7 Indexing an Attribute by Using ldapmodify

You can use the autocatalog feature, which is enabled by default in fresh installs. You can also enable it if you have upgraded from a previous release. When this feature is enabled, Oracle Internet Directory automatically invokes the `catalog` command to index attributes when you search for them. If the autocatalog feature is not enabled, and you want to use previously uncataloged attributes in search filters, you must add them to the catalog entry, as in previous releases.

You can add an attribute to the catalog entry by using `ldapmodify`.

To add an attribute, import an LDIF file by using `ldapmodify`. For example, to index the attribute `displayName`, import the following LDIF file by using ldapmodify:

```
dn: cn=catalogs
changetype: modify
add: orclindexedattribute
orclindexedattribute: displayName
```

Type a command similar to the following at the system prompt:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

To index the attribute, the `ldapmodify` command invokes the Catalog Management tool, `catalog`. For information about that tool, see Creating and Dropping Indexes from Existing Attributes by Using catalog.

21.3.8 Dropping an Index from an Attribute by Using ldapmodify

To drop an index from an attribute by using `ldapmodify`, specify `delete` in the LDIF file. For example:

```
dn: cn=catalogs
changetype: modify
delete: orclindexedattribute
orclindexedattribute: displayName
```

> **See Also:**
>
> The `ldapmodify` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

21.3.9 Indexing an Attribute by Using the Catalog Management Tool

You can use the Catalog Management tool instead of `ldapmodify` to index an attribute and to drop an index from an attribute. See Creating and Dropping Indexes from Existing Attributes by Using catalog.

> **See Also:**
>
> The `catalog` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

> **Note:**
>
> Unless you are absolutely sure that the indexes were not created by the base schema that was installed with Oracle Internet Directory, be careful not to use the `catalog delete=T` option to remove indexes from attributes. Removing indexes from base schema attributes can adversely impact the operation of Oracle Internet Directory.

21.3.10 Adding a New Attribute With Attribute Aliases by Using the Command Line

You add, modify, or delete attribute aliases by creating an LDIF file, then using `ldapmodify` with the following syntax:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

> **Note:**
>
> DN is not an attribute. You cannot define `dn` in the schema. Therefore you cannot create an alias for `dn`.

The following LDIF file adds the attribute `myattr` with the attribute aliases `myalias1` and `myalias2`:

```
dn: cn=subschemasubentry
changetype: modify
add: attributetypes
attributetypes: ( 1.2.3.4.5 NAME ( 'myattr' 'myalias1' 'myalias2' ) SYNTAX
                  '1.3.6.1.4.1.1466.115.121.1.38' )
```

21.3.11 Adding or Modifying Attribute Aliases in Existing Attributes by Using the Command Line

The following LDIF file adds the attribute aliases `surname` and `mysurName` to the existing attribute `sn`:

```
dn: cn=subschemasubentry
changetype: modify
delete: attributeTypes
attributeTypes: ( 2.5.4.4 NAME 'sn' SUP name )
-
add: attributeTypes
attributeTypes: ( 2.5.4.4 NAME ( 'sn' 'surname' 'mysurName' ) SUP name )
```

Type a command similar to the following at the system prompt:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

21.3.12 Deleting Attribute Aliases by Using the Command Line

Use the `ldapmodify` command to delete attribute aliases. The following LDIF file deletes the attribute alias `mysurName` but not the attribute alias `surName` from the attribute `sn`:

```
dn: cn=subschemasubentry
changetype: modify
delete: attributeTypes
attributeTypes: ( 2.5.4.4 NAME ( 'sn' 'surname' 'mysurName' ) SUP name )
-
add: attributeTypes
attributeTypes: ( 2.5.4.4 NAME ( 'sn' 'surname' ) SUP name )
```

The following LDIF file deletes both attribute aliases, `surname` and `mysurName`, from the attribute `sn`:

```
dn: cn=subschemasubentry
changetype: modify
delete: attributeTypes
attributeTypes: ( 2.5.4.4 NAME ( 'sn' 'surname' 'mysurName' ) SUP name )
-
add: attributeTypes
attributeTypes: ( 2.5.4.4 NAME 'sn' SUP name )
```

21.3.13 Using Attribute Aliases with LDAP Commands

After you define attribute aliases in the LDAP schema, users can substitute the aliases for attribute names in LDAP operations. The following examples show the commands to use and the results to expect:

Table 21-1 shows the aliases used in the examples and the attributes names they represent:

Table 21-1 Attribute Aliases Used in Examples

Alias	Attribute Name
`userid`	`uid`
`organizationalunit`	`ou`

Table 21-1 (Cont.) Attribute Aliases Used in Examples

Alias	Attribute Name
country	c
organization	o
surname	sn
commonname	cn
phone	telephonenumber

- Using Attribute Aliases with ldapsearch
- Using Attribute Aliases with ldapadd
- Using Attribute Aliases with ldapmodify
- Using Attribute Aliases with ldapdelete
- Using Attribute Aliases with ldapmoddn

21.3.13.1 Using Attribute Aliases with ldapsearch

The LDAP server recognizes attribute aliases in the search filter string, in the base DN, and in the required attributes list in `ldapsearch` operations. The search result contains the actual attribute names, unless the user explicitly asks for an alias using the required attributes list. For example, suppose the user specifies the following search, using the aliases `organizationalUnit`, `country`, and `organization` in the base DN and the alias `surname` in the filter string:

```
ldapsearch -p 3060 -h myhost \
   -b "organizationalUnit=dev,country=us,organization=myorg" \
   -s sub "surname=brown"
```

The search returns a result similar to this:

```
uid=mbrown,ou=dev,c=us,o=myorg
uid=mbrown
sn=Brown
cn=Mark Brown
telephonenumber;office=444006
telephonenumber;mobile=555006
objectclass=organizationalPerson
objectclass=top
objectclass=person
```

Now suppose the user specifically asks for the aliases `surname`, `commonname`, and `userid` by including them in a required attributes list, like this:

```
ldapsearch -p 3060 -h myhost \
   -b "organizationalUnit=dev,country=us,organization=myorg" \
   -s sub "surname=brown" surname commonname userid phone
```

Because the user specifically included the aliases, the search returns a result similar to this:

```
uid=mbrown,ou=dev,c=us,o=myorg
surname=Brown
commonname=Mark Brown
userid=mbrown
```

```
phone;office=444006
phone;mobile=555006
```

21.3.13.2 Using Attribute Aliases with ldapadd

The LDAP server recognizes attribute aliases in place of attribute names during the add operation. When the LDAP server stores the entry, it replaces the alias with the actual attribute name.

The command-line format is:

```
ldapadd -h host -p port -D cn=orcladmin -q -f ldif_file_name
```

The user could provide an LDIF file like this:

```
dn: userid=mbrown,organizationalUnit=dev,country=us,organization=myorg
objectclass: account
objectclass: organizationalPerson
userID: mbrown
surname: Brown
commonName: Mark Brown
userpassword: password
phone;office: 444006
phone;mobile: 555006
```

The entry is stored as if the file contained the attribute names instead of the aliases. On subsequent LDAP searches, however, the DN is returned as it was entered when added or modified:

```
dn: userid=mbrown,organizationalUnit=dev,country=us,organization=myorg
```

This is standard behavior for LDAP search results. The DN is always returned with the same format that was used when the entry was created.

21.3.13.3 Using Attribute Aliases with ldapmodify

The LDAP server recognizes attribute aliases in place of attribute names during the modify operation.

The command-line format is:

```
ldapmodify -D "cn=orcladmin" -q -h host -p port -f ldif_file_name
```

The user could provide an LDIF file like this:

```
dn:
userid=mbrown,organizationalUnit=dev,country=us,organization=myorg
changetype: modify
replace: surname
surname: davis
```

The entry is stored as if the file contained the attribute names instead of the aliases. On subsequent LDAP searches, however, the DN is returned as it was entered when added or modified:

```
dn: userid=mbrown,organizationalUnit=dev,country=us,organization=myorg
```

This is standard behavior for LDAP search results. The DN is always returned with the same format that was used when the entry was created.

21.3.13.4 Using Attribute Aliases with ldapdelete

The LDAP server recognizes attribute aliases in the DN provided for delete operations. For example, suppose the user provides a request like this, with the aliases `userid`, `organizationalUnit`, `country`, and `organization` in the search filter:

```
ldapdelelete -D "cn=orcladmin" -q -p 3060 \
   -h myhost  "userid=mbrown,organizationalUnit=dev,country=us,organization=myorg"
```

The server deletes the entry as if the user had typed:

```
ldapdelelete -D "cn=orcladmin" -q -p 3060 \
   -h myhost  "uid=mbrown,ou=dev,c=us,o=myorg"
```

21.3.13.5 Using Attribute Aliases with ldapmoddn

The LDAP server recognizes attribute aliases in the DN, the new RDN, and the new parent DN options. For example, suppose the user types the command line:

```
ldapmoddn -D "cn=orcladmin" -q \
   -b "userid=mbrown,organizationalUnit=dev,country=us,organization=myorg"\
   -R "userid=mdavis"
```

The LDAP server interpret the command line as if the user had typed

```
ldapmoddn -D "cn=orcladmin" -q -b "uid=mbrown,ou=dev,c=us,o=myorg" \
   -R "uid=mdavis"
```

The entry is stored as if the file contained the attribute names instead of the aliases. On subsequent LDAP searches, however, the DN is returned as it was entered when added or modified:

```
dn: userid=mbrown,organizationalUnit=dev,country=us,organization=myorg
```

This is standard behavior for LDAP search results. The DN is always returned with the same format that was used when the entry was created.

21.3.14 Managing Content Rules by Using Command-Line Tools

The format of a content rule is:

```
DITContentRule  ::=  SEQUENCE  {

oids                              ALPHA-NUMERIC-OID,
structuralObjectClass             OBJECT-CLASS,
LABEL                             CONTENT-LABEL OPTIONAL,
auxiliaries                       SET (1..MAX) OF OBJECT-CLASS OPTIONAL,
mandatory                         SET (1..MAX) OF ATTRIBUTE OPTIONAL,
optional                          SET (1..MAX) OF ATTRIBUTE OPTIONAL,
precluded                         SET (1..MAX) OF ATTRIBUTE OPTIONAL
}
```

You can use `ldapmodify` with an LDIF file such as the following to add a content rule:

```
dn: cn=subschemasubentrychangetype: modify
add: contentrules
contentrules: (  2.16.840.1.113894.9999.1.1 OBJECTCLASS 'exampleObjClassName' LABEL abc
AUX exampleAuxObjClassName MUST businessCategory MAY ( description $ host ) )
```

The following section describes the Content Rule Parameters. Note that the attributes and object class names are case-sensitive.

- **oids**

  A unique object identifier (oids) for the content rule similar to the one for an object class or attribute definition. It must be a unique numeric value that begins with 2.16.840.1.113894 followed by .9999 or a site-specific prefix.

- **LABEL**

  The content label of the content rule as applied in the directory.

- **structuralObjectClass**

  The structural object class to which the content rule applies.

- **auxiliaries**

  The auxiliary object classes allowed for an entry to which the content rule applies.

- **mandatory**

  User attribute types contained in an entry to which the content rule applies. These are in addition to those mandatory attributes that the entry contains as a result of its association with its specified structural and auxiliary object classes.

- **optional**

  User attribute types that may be contained in an entry to which the content rule applies. These are in addition to those that the entry may contain as a result of its association with its specified structural and auxiliary object classes.

21.3.15 Viewing Matching Rules by Using ldapsearch

> **Note:**
>
> Matching rules cannot be modified.

To view matching rules, type:

```
ldapsearch -L -D "cn=orcladmin" -p port -q -b "cn=subschemasubentry" -s base
"objectclass=*" matchingrules
```

> **See Also:**
>
> The `ldapsearch` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

21.3.16 Viewing Syntaxes by Using ldapsearch

> **Note:**
>
> Syntaxes cannot be modified.

To view syntaxes, type:

```
ldapsearch -L -D "cn=orcladmin" -p port -q -b "cn=subschemasubentry" -s base
"objectclass=*" ldapsyntaxes
```

> **See Also:**
>
> The `ldapsearch` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

22

Configuring Referential Integrity

This chapter describes how to enable, disable, and configure referential integrity in Oracle Internet Directory using Oracle Enterprise Manager Fusion Middleware Control and LDAP command-line utilities. It also describes how to detect and correct referential integrity violations.

- Introduction to Configuring Referential Integrity
- Enabling Referential Integrity Using Fusion Middleware Control
- Disabling Referential Integrity Using Fusion Middleware Control
- Enabling Referential Integrity Using the Command Line
- Configuring Specific Attributes for Referential Integrity by Using the Command Line
- Disabling Referential Integrity by Using the Command Line
- Detecting and Correcting Referential Integrity Violations

22.1 Introduction to Configuring Referential Integrity

Referential integrity is the process of maintaining consistent relationships among sets of data.

If referential Integrity is enabled in Oracle Internet Directory, whenever you update an entry in the directory, the server also updates other entries that refer to that entry. For example, if you remove a user's entry from the directory, and the user is a member of a group, the server also removes the user from the group. If referential integrity is not enabled, the user remains a member of the group until manually removed. Referential integrity is not enabled by default.

> **Note:**
>
> Disable referential integrity during the replication bootstrapping process. If referential integrity is enabled, bootstrapping fails.

Referential integrity takes effect in two situations:

- Delete–When an entry is deleted, all the DN attributes that refer to this entry DN are removed.
- Modify–When an entry's DN is modified (renamed), all the attributes that refer to this entry DN are modified.

The Oracle Internet Directory server can enforce referential integrity. For every LDAP add, modify, delete, and rename operation, the server monitors the request and updates the necessary DN references.

Two configuration parameters control referential integrity: `orclRIenabled` and `orclRIattr`.

- The parameter `orclRIenabled` controls the referential integrity level. Values for `orclRIenabled` are:

  - 0–Referential integrity is disabled
  - 1–Referential integrity is enabled for `member` and `uniquemember` attributes only.
  - 2–Referential Integrity is enabled for a list of DN syntax attributes as specified in orclRIattr and for attributes `member` and `uniquemember`.
- When `orclRIenabled` is set to 2, the value of the parameter `orclRIattr` takes effect. The value of `orclRIattr` is a list of referential integrity-enabled attributes.

If referential integrity is enabled, it is strictly enforced. For example, you cannot add a group entry whose `member` or `uniquemember` attributes are not currently part of the DIT.

22.2 Enabling Referential Integrity Using Fusion Middleware Control

You can enable referential integrity by using Oracle Enterprise Fusion Middleware Control.

To configure and enable referential integrity by using Oracle Enterprise Manager Fusion Middleware Control, perform the following steps:

1. Select **Administration**, then **Shared Properties** from the **Oracle Internet Directory** menu, then select **General**.
2. Select a value from the Referential Integrity list:
   - Enabled for GroupofNames and GroupofUniqueNames
   - Enabled for GroupofNames, GroupofUniqueNames, and configured DN attributes
3. Choose **Apply**.

22.3 Disabling Referential Integrity Using Fusion Middleware Control

You can disable referential integrity by using Oracle Enterprise Manager Fusion Middleware Control.

To disable referential integrity by using Oracle Enterprise Manager Fusion Middleware Control, perform the following steps:

1. Select **Administration**, then **Shared Properties** from the **Oracle Internet Directory** menu, then select **General**.
2. Select **Disabled** from the Enable Referential Integrity list.

22.4 Enabling Referential Integrity Using the Command Line

This section gives a description of how to enable referential integrity using command line utility.

You enable referential integrity in the directory by using `ldapmodify` to change the value of the parameter `orclRIenabled` in the DSA Configuration entry:

```
cn=dsaconfig,cn=configsets,cn=oracle internet directory.
```

You can set the value to either 1 or 2.

Setting a value of `1` enables referential integrity for `GroupofNames` and `GroupofUniqueNames`.

Setting a value of 2 for `orclRIenabled` enables referential integrity for `GroupofNames` and `GroupofUniqueNames` and for specific configured attributes. The next section describes configuring specific attributes.

For example, you would use a command line such as:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

with an LDIF file such as:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype: modify
replace: orclRIenabled
orclRIenabled: 2
```

Changes to `orclRIenabled` take effect immediately.

22.5 Configuring Specific Attributes for Referential Integrity by Using the Command Line

When `orclRIenabled` is set to `2`, referential integrity is enabled for `GroupofNames`, `GroupofUniqueNames`, and for specific configured attributes.

You configure specific attributes for referential integrity by using `catalog` with the arguments `rienable=TRUE`, `add=true`, and `attribute=name_of_attribute`. This adds the attribute to `orclRIattr`, which contains the list of DN syntax attributes to which referential integrity applies. You remove an attribute from referential integrity by using `catalog` with the arguments `rienable=TRUE`, `delete=true`, and `attribute=name_of_attribute`. This removes the attribute from `orclRIattr`.

> **Note:**
>
> - You cannot change the value of `orclRIattr` by using `ldapmodify`. You must use the `catalog` command.
> - Remember that the `DOMAIN_HOME` environment variable must be set when you use `catalog`.

This example enables referential integrity for the attribute `manager`.

```
catalog connect="connect_str" add=true rienable="TRUE" attribute="manager"
```

This example disables referential integrity for the attribute `manager`.

```
catalog connect="connect_str" delete=true rienable="TRUE" attribute="manager"
```

22.6 Disabling Referential Integrity by Using the Command Line

You can disable referential integrity by using command line utility.

To disable referential integrity in the directory, set the value of `orclRIenabled` to `0` in the DSA Configuration entry:

```
cn=dsaconfig,cn=configsets,cn=oracle internet directory.
```

22.7 Detecting and Correcting Referential Integrity Violations

When you try to enable referential integrity, if there are underlying violations in the DIT, you get an error.

You must run the `oiddiag` tool to look at the violations, rectify them, and then enable referential integrity. The `oiddiag` tool has an option, `OidDiagDC10`, to report all the referential integrity violations. in LDIF format. That LDIF file can be used with `ldapmodify` tool to fix all reported entries. The steps are as follows:

1. Run `oiddiag` with the option `listdiags=true`. The default output file is *`$DOMAIN_HOME`*`/tools/OID/logs/oiddiag.txt`.
2. Edit the output file, `oiddiag.txt` so that it contains only the line:

   ```
   oracle.ldap.oiddiag.dc.OidDiagDC10
   ```
3. Run `oiddiag` with the option `collect_sub=true`

> **See Also:**
>
> - The `oiddiag` command reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*
> - The `oiddiag` usage message. Type:
>
>   ```
>   oiddiag -help
>   ```

> **Note:**
>
> On Windows, the filename of the `oiddiag` command is `oiddiag.bat`.

23
Managing Auditing

This chapter describes how to manage auditing for information specific to Oracle Internet Directory using Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Scripting Tool (WLST), and LDAP command-line utilities.
See the Audit Administration Tasks in *Securing Applications with Oracle Platform Security Services*.

Before reading this chapter, read Introduction to Oracle Fusion Middleware Audit Service in *Securing Applications with Oracle Platform Security Services*.

The new Oracle Internet Directory audit framework has the following advantages:

- It uses the same record format as other Oracle Application Server components.
- Records are stored in Oracle Database tables for better performance and security.
- Records can be stored in Audit Vault for increased security.
- As administrator, you can configure the type of information captured in the audit records by using Enterprise Manager.
- Configuration changes are effective immediately.
- An administrator can view audit records:
  - In Enterprise Manager
  - In summary reports based on XML Publisher

This chapter includes the following sections:

- Introduction to Auditing
- Managing Auditing Using Fusion Middleware Control
- Managing Auditing Using WLST
- Managing Auditing from the Command Line

23.1 Introduction to Auditing

Auditing is the process that collects and stores information about security requests and the outcome of those requests, thus providing an electronic trail of selected system activity for non-repudiation purposes.

Auditing can be configured to track particular security events and management operations based on specific audit criteria. Audit records are kept in a centralized repository (LDAP, database, or file) that allows the creation, viewing, and storage of audit reports. The Oracle Internet Directory uses an audit framework that is integrated with Oracle Fusion Middleware. Oracle Internet Directory uses this framework to audit its critical security related operations. The features of the framework are:

- APIs for collecting audit information from AS components
- Common audit record format to be used by all AS components

- Audit repository database that collects audit records produced by components in the enterprise. (The customer also has an option to use the Audit vault as a repository)
- Administrative interface for controlling the type of information captured by the audit facility.

All audit configuration performed by the instance administrator is audited. This cannot be disabled.

> **See Also:**
>
> Configuring and Managing Auditing in *Securing Applications with Oracle Platform Security Services* for information about configuring the audit repository and audit filters.

This introduction contains the following topics:

- Configuring the Audit Store
- Oracle Internet Directory Audit Configuration
- Replication and Oracle Directory Integration Platform Audit Configuration
- Audit Record Fields
- Audit Record Storage
- Generating Audit Reports

23.1.1 Configuring the Audit Store

You must configure an audit store to ensure that audit records are saved in a database.

See Configuring and Managing Auditing Chapter in *Securing Applications with Oracle Platform Security Services* for complete coverage of Audit Administration Tasks, including:

- Managing the Audit Store
- Advanced Management of Database Store

23.1.2 Oracle Internet Directory Audit Configuration

Audit configuration for Oracle Internet Directory consists of three attributes of the instance-specific entry:

The three attributes of the instance-specific entry are:

`cn=componentname,cn=osdldapd,cn=subconfigsubentry`

Table 23-1 describes these attributes.

Table 23-1 Oracle Internet Directory Audit Configuration Attributes

Attribute	Description
`orclAudFilterPreset`	Presets are `None`, `Low`, `Medium`, `All`, and `Custom`, where `Low` specifies Account Management, Change Password and ModifyDataItemAttributes events and `Medium` specifies all events in `Low` plus Failed authentication events.

Table 23-1 (Cont.) Oracle Internet Directory Audit Configuration Attributes

Attribute	Description

orclAudCustEvents	A comma-separated list of events and category names to be audited. Examples include:
	`Authentication.SUCCESSESONLY,`
	`Authorization(Permission -eq 'CSFPermission")`
	Custom events are only applicable when `orclAudFilterPreset` is `Custom`.

orclAudSplUsers	A comma separated list of users for whom auditing is always enabled, even if `orclAudFilterPreset` is `None`. For example: `cn=orcladmin.`

For more information, see the Configuring and Managing Auditing in *Securing Applications with Oracle Platform Security Services*.

23.1.3 Replication and Oracle Directory Integration Platform Audit Configuration

Replication and Oracle Directory Integration Platform auditing can be enabled by changing the value of the attribute `orclextconfflag` in the instance-specific configuration entry.

The default value is `3`, which disables both replication and Oracle Directory Integration Platform auditing. To enable both, change it to `7`. This is the only change you can make to `orclextconfflag`, which is otherwise an internal attribute.

See Enabling Replication and Oracle Directory Integration Platform Auditing.

23.1.4 Audit Record Fields

The audit record consists of many fields.

Audit records contain the following fields:

- Event category–the class of event, such as authentication or authorization.
- Event name
- Initiator–the user who initiates the operation
- Status–success or failure
- Authentication method
- Session ID–Connection ID
- Target–the user on whom the operation is performed
- Event date and time
- Remote IP–source IP address of client
- Component type–`OID`
- ECID

- Resource–entry or attribute on which operation is performed.

23.1.5 Audit Record Storage

Audit information is held temporarily in a location called a busstop before it is written to its final location.

The file is in the directory *ORACLE_INSTANCE*/auditlogs/*componentType*/*componentName*.

Audit files are permanently stored in either XML files or a database. XML files are the default storage mechanism for audit records. There is one XML repository for each Oracle instance. Audit records generated for all components running in a givenOracle instance are stored in the same repository. If using a database repository, audit records generated by all components in all Oracle instances in the domain are stored in the same repository.

23.1.6 Generating Audit Reports

This section has reference about how to generate audit reports.

See *Securing Applications with Oracle Platform Security Services* chapter on audit analysis and reporting for information about generating audit reports. There are Oracle Internet Directory examples in Configuring and Managing Auditing in *Oracle Fusion Middleware Application Security Guide*.

23.2 Managing Auditing Using Fusion Middleware Control

The Oracle Fusion Middleware Audit Framework provides a centralized audit framework for Oracle middleware products, including system components such as Oracle Internet Directory.

You can use Oracle Enterprise Manager Fusion Middleware Control to manage auditing. The interface is basically the same for all Oracle Fusion Middleware components, as documented in the Managing the Audit Data Store in *Securing Applications with Oracle Platform Security Services*.

To manage Oracle Internet Directory auditing.

1. Login to Oracle Enterprise Manager Fusion Middleware Control.
2. From the Oracle Internet Directory menu, select **Security**, then **Audit Policy Settings**.
3. From the Audit Policy list, select **Custom** to configure your own filters, or one of the filter presets, **None**, **Low**, or **Medium**. (You cannot set `All` from Fusion Middleware Control.)
4. If you want to audit only failures, click Select Failures Only. (You can only do this if you selected **Custom** in the previous step.
5. To configure a filter, click the **Edit** icon next to its name. The Edit Filter dialog for the filter appears.
6. Specify the filter condition using the buttons, selections from the menus, and strings that you enter. Condition subjects include HostID, HostNwaddr, InitiatorDN, TargetDN, Initiator, Remote IP, and Roles. Condition tests include -contains, -contains_case, endswith, endswith_case, -eq, -ne, -startswith, and -startswith-case. Enter values for the tests as strings. Parentheses are used for grouping and AND and OR for combining.
7. To add a condition, click the **Add** icon.
8. When you have completed the filter, click **Apply** to save the changes or **Revert** to discard the changes.

Oracle Internet Directory stores its audit configuration in the three instance-specific configuration entry attributes described in Table 23-1. The correspondence between the fields on the Audit Policy Page and the attributes is shown in Table 23-2.

Table 23-2 Audit Configuration Attributes in Fusion Middleware Control

Field or Heading	Configuration attribute
Audit Policy	`orclAudFilterPreset`
Name, Select Failures Only, Enable Audit, Filter	`orclAudCustEvents`
Users to always audit	`orclAudSplUsers`

23.2.1 Auditing Oracle Internet Directory Sensitive Data Attributes

Using the Oracle Fusion Middleware Audit Framework, you can define a custom Oracle Internet Directory audit policy to monitor the attributes associated with sensitive data such as access control, user credentials, and configuration.

For example, to audit changes to access control policy points (ACPs), you can configure an audit policy to capture the `ModifyDataItemAttributes` event type for attributes such as `orclaci` and `orclentrylevelaci`.

The `ModifyDataItemAttributes` event type is generated by `ldapmodify` operations. The initiator attribute for this event is the DN of the user performing the LDAP operation, and resource attribute is the entry DN of the LDAP attribute on which the operation is performed.

By capturing the `ModifyDataItemAttributes` event type, you can monitor all ACP changes to attributes. For example, you can determine if any changes are made to the `orclaci` and `orclentrylevelaci` attributes.

To create an audit policy to monitor changes to the `orclaci` and `orclentrylevelaci` attributes:

1. Login to Oracle Enterprise Manager Fusion Middleware Control.
2. In the left panel, right-click the Oracle Internet Directory instance you want to audit. For example: `oid1`
3. From the Oracle Internet Directory component menu, navigate to **Security** and then **Audit Policy**. The Audit Policy Settings page appears.
4. From the drop-down list for Audit Level, select **Custom**.
5. Check **Enable Audit** for the **DataAccess** event category and the **ModifyDataItemAttributes** event type.
6. Depending on whether you want to audit successful or failing changes (or both), click the appropriate check box in the Enable Audit column.
7. To add a filter for the audit policy, click the **Edit Filter** pencil icon and then configure the filter:
   a. Set the Condition to **Resource**.
   b. Set the operator to **-eq**.
   c. Specify the Resource attribute as **orclaci**.
   d. Click the **Add** icon. Your filter should be:

```
Resource -eq "orclaci"
```

e. To add the `orclentrylevelaci` attribute to the filter, click **OR**, define a new condition (**Resource**, **-eq**, **orclentrylevelaci**), and then click **Add**. Your filter should now be:

```
Resource -eq "orclaci" -or Resource -eq "orclentrylevelaci"
```

8. Click **Apply** to save the audit policy (or **Revert** to discard it.)

If required by your deployment, you can add other attributes to this policy, or create other policies to audit attributes, operations, and activities. For more information, see Manage Audit Policies for System Components with Fusion Middleware Control in the *Securing Applications with Oracle Platform Security Services*.

23.3 Managing Auditing Using WLST

You can use `wlst` to manage auditing.

See Manage Audit Policies with WLST in *Securing Applications with Oracle Platform Security Services*. You use the commands `getAuditPolicy()`, `setAuditPolicy()`, or `listAuditEvents()`.

For component that manage their audit policy locally, such as Oracle Internet Directory, you must include an MBean name as an argument to the command. The name for an Audit MBean is of the form:

```
oracle.as.management.mbeans.register:type=component.auditconfig,name=auditconfig1,instance=INSTANCE,component=COMPONENT_NAME
```

For example:

```
oracle.as.management.mbeans.register:type=component.auditconfig,name=auditconfig1,instance=instance1,component=oid1
```

Another `wlst` command you must use is `invoke()`. As described in Managing System Configuration Attributes by Using WLST, before you make any changes to attributes, you must ensure that the MBean has the current server configuration. To do that, you must use the `invoke()` command to load the configuration from Oracle Internet Directory server to the mbean. After making changes, you must use the `invoke()` command to save the MBean configuration to the Oracle Internet Directory server. In order to use `invoke()` in this way, you must navigate to the Root Proxy MBean in the tree. The name for a Root Proxy MBean is of the form:

```
oracle.as.management.mbeans.register:type=component,name=COMPONENT_NAME,instance=INSTANCE
```

For example:

```
oracle.as.management.mbeans.register:type=component,name=oid1,instance=instance1
```

Here is an example of a `wlst` session using `setAuditPolicy()` and `invoke()`:

```
$ORACLE_HOME/oracle_common/common/bin/wlst.sh
connect('username', 'password', 't3://localhost:7001')
custom()
cd('oracle.as.management.mbeans.register')
cd('oracle.as.management.mbeans.register:type=component,name=oid1,instance=instance1')
invoke('load',jarray.array([],java.lang.Object),jarray.array([],java.lang.String))
setAuditPolicy(filterPreset='None',
 on='oracle.as.management.mbeans.register:type=component.auditconfig,
 name=auditconfig1,instance=instance1,component=oid1')
invoke('save',jarray.array([],java.lang.Object),jarray.array([],java.lang.String))
```

23.4 Managing Auditing from the Command Line

This section helps you understand how to use LDAP tools to manage auditing:

Refer to the following topics:

- Viewing Audit Configuration from the Command Line
- Configuring Oracle Internet Directory Auditing from the Command Line
- Enabling Replication and Oracle Directory Integration Platform Auditing

23.4.1 Viewing Audit Configuration from the Command Line

You can use `ldapsearch` to view audit configuration.

For example:

```
ldapsearch -p 3060 -h myhost.example.com -D cn=orcladmin -q \
  -b "cn=oid1,cn=osdldapd,cn=subconfigsubentry" \
  -s base "objectclass=*" > /tmp/oid1-config.txt
grep orclaud oid1-config.txt
orclaudsplusers=cn=orcladmin
orclaudcustevents=UserLogin.FAILURESONLY, UserLogout, CheckAuthorization,
 ModifyDataItemAttributes, CompareDataItemAttributes, ChangePassword.FAILURESONLY
orclaudfilterpreset=custom
```

23.4.2 Configuring Oracle Internet Directory Auditing from the Command Line

You can use `ldapmodify` commands to manage auditing.

You must create an LDIF file to make the required changes to the attributes orclAudFilterPreset, orclAudCustEvents, and orclAudSplUsers.

The command is:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

For example to enable auditing for user login events only, use this LDIF file with the preceding `ldapmodify` command:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclaudFilterPreset
orclaudFilterPreset: Custom
-
replace: orclaudcustevents
orclaudcustevents: UserLogin
```

For more information, see the Manage Audit Policies with WLST in *Securing Applications with Oracle Platform Security Services*.

23.4.3 Enabling Replication and Oracle Directory Integration Platform Auditing

You can use LDIF file to enable both replication and Oracle Directory Integration Platform auditing.

The following LDIF file enables both replication and Oracle Directory Integration Platform auditing.

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclextconfflag
orclextconfflag: 7
```

The following LDIF file disables both:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclextconfflag
orclextconfflag: 3
```

Use a command line similar to this:

```
ldapmodify -h host -p port -D "cn=orcladmin" -q -f ldiffile
```

24
Managing Logging

This chapter describes logging by Oracle Internet Directory, including how to view log files and configure debug logging using Oracle Enterprise Manager Fusion Middleware Control and LDAP command-line utilities.
For general information about logging in Oracle Fusion Middleware, see Managing Log Files and Diagnostic Data in *Administering Oracle Fusion Middleware*.

This chapter includes the following sections:

- Introduction to Logging
- Managing Logging Using Fusion Middleware Control
- Managing Logging from the Command Line

24.1 Introduction to Logging

Like other Oracle Fusion Middleware components, Oracle Internet Directory writes diagnostic log files in the Oracle Diagnostic Logging (ODL) format.

> **See Also:**
>
> Managing Log Files and Diagnostic Data in *Administering Oracle Fusion Middleware* for information about ODL.

- Oracle Internet Directory File Locations
- Features of Oracle Internet Directory Debug Logging
- Understanding Log Messages

24.1.1 Oracle Internet Directory File Locations

Oracle Internet Directory tools and servers output their log and trace information to log files in the *$DOMAIN_HOME*.

Table 24-1 lists each component and the location of its corresponding log file.

Table 24-1 Oracle Internet Directory Log File Locations

Tool or Server Name	Log File Name
Bulk Loader (`bulkload`)	*$DOMAIN_HOME*`/tools/OID/logs/bulkload.log`
Bulk Modifier (`bulkmodify`)	*$DOMAIN_HOME*`/tools/OID/logs/bulkmodify.log`
Bulk Delete Tool (`bulkdelete`)	*$DOMAIN_HOME*`/tools/OID/logs/bulkdelete.log`

Table 24-1 (Cont.) Oracle Internet Directory Log File Locations

Tool or Server Name	Log File Name
Catalog Management Tool (`catalog`)	*$DOMAIN_HOME*`/tools/OID/logs/catalog.log`
Data Export Tool (`ldifwrite`)	*$DOMAIN_HOME*`/tools/OID/logs/ldifwrite.log`
Directory replication server (`oidrepld`)	*$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidrepld-`*XXXX*`.log` where *XXXX* is a number from 0000 to `orclmaxlogfiles`.
Directory server (`oidldapd`)	*$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidldapd01s`*PID*`-`*XXXX*`.log` where: • 01 is the instance number, which is 01 by default • s stands for server • PID is the server process identifier • *XXXX* is a number from 0000 to `orclmaxlogfiles` *$DOMAIN_HOME/servers/OID/logs/componentName/oidstackInstNumberPID.log* **Note:** The *oidstackInstNumber* log files pertain to SIGSEGV/SIGBUS tracing. Also, empty files with this name are created during directory instance startup, and can be ignored.
LDAP dispatcher (`oidldapd`)	*$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidldapd01-`*XXXX*`.log` where 01 is the instance number, which defaults to 01, and *XXXX* is a number from 0000 to `orclmaxlogfiles`.
OID Monitor (OIDMON)	*$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidmon-`*XXXX*`.log` where *XXXX* is a number from 0000 to `orclmaxlogfiles`. **Note:** If the log file size increases over `1` MB regardless the settings specified for `orclmaxlogfilesize` then OID Monitor (OIDMON) and OIDDISPD (dispatcher) log files are rotated after restarting Oracle Internet Directory.

24.1.2 Features of Oracle Internet Directory Debug Logging

Oracle Internet Directory enables you to view logging information.

Oracle Internet Directory enables you to:

- View logging information for the directory server, the directory replication server, and the directory integration server
- Set the logging level
- Specify one or more operations for which you want logging to occur

- Search messages in a standard format to determine remedial action for fatal and serious errors
- View trace messages according to their severity and order of importance
- Diagnose Oracle Internet Directory components by examining trace messages with relevant information about, for example, entry DN, ACP evaluation, and the context of an operation

24.1.3 Understanding Log Messages

This section discusses log messages—those associated with specified LDAP operations and those not. It provides an example of a trace log and explains how to interpret it.

Like other Oracle Fusion Middleware components, Oracle Internet Directory writes diagnostic log files in the Oracle Diagnostic Logging (ODL) format. The *Administering Oracle Fusion Middleware* describes ODL format.

- Log Messages for Specified LDAP Operations
- Log Messages Not Associated with Specified LDAP Operations
- Example for Trace Messages in Oracle Internet Directory Server Log File

24.1.3.1 Log Messages for Specified LDAP Operations

Log messages for a specified operation are stored as a trace object. This object tracks the operation from start to finish across the various Oracle Internet Directory modules. It is entered in the log file when one of the following occur:

- An LDAP operation completes
- A high priority message is logged
- The trace messages buffer is full

Each thread has one contiguous block of information for each operation, and that block is clearly delimited. This makes it easy, in a shared server environment, to follow the messages of different threads, operations, and connections.

If, because of an internal message buffer overflow, a single trace object cannot contain all the information about an operation, then the information is distributed among multiple trace objects. Each distributed piece of information is clearly delimited and has a common header. To track the progress of the operation, you follow the trace objects and their common header to the end, which is marked with the trace message "Operation Complete".

24.1.3.2 Log Messages Not Associated with Specified LDAP Operations

Messages not associated with any LDAP operation are represented in a simple format, which is not object-based. It is entered in the log file when either the operation completes or a high priority message is encountered.

A thread that does not perform an operation logs only trace messages. Its header contains the date, time, and the thread identifier. It does not contain the Execution Context ID (ECID) or connection and operation-related information.

A trace object starts with the keyword `BEGIN` and ends with the keyword `END`.

24.1.3.3 Example for Trace Messages in Oracle Internet Directory Server Log File

```
[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: Starting up the OiD Server, on node
srvhst.us.abccorp.com.


[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: Oid Server Connected to DB store via
inst1 connect string.

[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: Loading Root DSE ...

[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: Loading subschema subentry ...

[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: Loading catalog entry ...

[2008-11-14T15:28:01-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 0] Main:: OiD LDAP server started.

[2008-11-14T15:28:02-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 2] ServerDispatcher : Thread Started

[2008-11-14T15:28:02-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 1] ServerDispatcher : Thread Started

[2008-11-14T15:28:02-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 3] ServerWorker (REG): Thread Started

[2008-11-14T15:28:02-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 4] ServerWorker (REG): Thread Started

[2008-11-14T15:28:02-08:00] [OID] [NOTIFICATION:16] [] [OIDLDAPD] [host:
srvhst.us.abccorp.com] [pid: 7043] [tid: 5] ServerWorker (SPW): Thread Started


[2008-11-14T15:28:47-08:00] [OID] [TRACE:16] [] [OIDLDAPD] [host: srvhst.us.abccorp.com]
[pid: 7043] [tid: 3] [ecid: 004MuuNFY7UCknT6uBU4UH0001i30000Ee,0] ServerWorker (REG):[[
BEGIN
 ConnID:87 mesgID:2 OpID:1  OpName:bind ConnIP:170.90.11.210 ConnDN:cn=orcladmin
15:28:47-08:00 * gslfbiADoBind * Entry
15:28:47-08:00 * gslfbiGetControlInfo * Entry
15:28:47-08:00 * gslfbiGetControlInfo * Exit
15:28:47-08:00 * gslfbidbDoBind *  Version=3 BIND dn="cn=orcladmin" method=128
15:28:47-08:00 * gslsbnrNormalizeString * String to Normalize: "orcladmin"
15:28:47-08:01 * gslsbnrNormalizeString * Normalized value: "orcladmin"
15:28:47-08:01 * gslfrsBSendLdapResult * Entry
15:28:47-08:01 * gslfrsASendLdapResult2 * Entry
15:28:47-08:01 * sgslunwWrite * Entry
15:28:47-08:01 * sgslunwWrite * Exit
15:28:47-08:01 * gslfrsASendLdapResult2 * Exit
15:28:47-08:01 * gslfrsBSendLdapResult * Exit
15:28:47-08:01 * gslfbiADoBind * Exit
TOTAL Worker time :  4402  micro sec
END
]]
```

```
[2008-11-14T15:28:56-08:01] [OID] [TRACE:16] [] [OIDLDAPD] [host: srvhst.us.abccorp.com]
[pid: 7043] [tid: 4] [ecid: 004MuuNqbefCknT6uBU4UH0001i30000Lf,0] ServerWorker (REG):[[
BEGIN
 ConnID:126 mesgID:1 OpID:0  OpName:bind ConnIP:170.90.11.210 ConnDN:Anonymous
15:28:56-08:01 * gslfbiADoBind * Entry
15:28:56-08:01 * gslfbiGetControlInfo * Entry
15:28:56-08:01 * gslfbiGetControlInfo * Exit
15:28:56-08:01 * gslfbidbDoBind *  Version=3 BIND dn="" method=128
15:28:56-08:01 * gslfrsBSendLdapResult * Entry
15:28:56-08:01 * gslfrsASendLdapResult2 * Entry
15:28:56-08:01 * sgslunwWrite * Entry
15:28:56-08:02 * sgslunwWrite * Exit
15:28:56-08:02 * gslfrsASendLdapResult2 * Exit
15:28:56-08:02 * gslfrsBSendLdapResult * Exit
15:28:56-08:02 * gslfbiADoBind * Exit
TOTAL Worker time :  2591  micro sec
END
]]
```

24.2 Managing Logging Using Fusion Middleware Control

You can view log files and configure debug logging with Oracle Enterprise Manager Fusion Middleware Control.

For more information, refer to the following sections:

- Viewing Log Files Using Fusion Middleware Control
- Configuring Logging Using Fusion Middleware Control

24.2.1 Viewing Log Files Using Fusion Middleware Control

You can view log files using Fusion Middleware Control.

To view the log files using the Fusion Middleware Control:

1. From the Oracle Internet Directory menu, select **Logs**, then **View Log Messages**. The Log Messages page appears.
2. Select the date range for the logs you want to view. You can select **Most Recent**, by minutes, hours or days. Alternatively, you can select a **Time Interval** and specify the date and time to start and end.
3. Select the Message Types you want to view.
4. Specify the **Maximum Rows Displayed**.
5. From the **View** list, select **Columns** to change the columns shown. Select **Reorder Columns** to change the order of the columns.
6. Within each column, you can toggle between ascending and descending order by choosing the up or down arrow in the column header.
7. From the Show list, choose whether to show all messages, a summary by message type, or a summary by message id.
8. To perform a specific search, choose **Add Fields** and add fields to search on. For each field, select a criterion from the list, then enter text into the box. Choose the red **X** to delete a field. Choose **Add Fields** to add additional fields. When you have finished adding criteria, choose **Search**.
9. Use the **Broaden Target Scope** list to view messages for the Domain.

10. Choose **Export Messages to File** to export the log messages to a file as XML, text, or comma-separated list.
11. Click **Target Log Files** to view information about individual log files.
12. You can indicate when to refresh the view. Select Manual Refresh, 30-Second Refresh, or One Minute Refresh from the list on the upper right.
13. Use the **View** list to change the columns listed or to reorder columns.
14. Use the **Show** list to change the grouping of messages.
15. Collapse the Search label to view only the list of log messages.
16. To view the contents of a log file, double click the file name in the Log File column. The View Log File: *filename* page is displayed. You can use the up and down arrows in the Time, Message Type, and Message ID to reorder the records in the file.

24.2.2 Configuring Logging Using Fusion Middleware Control

You can configure logging using Fusion Middleware Control.

Table 24-2 Configuration Attributes on Server Properties Page, Logging Tab

Field or Heading	Configuration Attribute
Debug Level	`orcldebugflag`
Operations Enabled for Debug	`orcldebugop`
Maximum Log File Size (MB)	`orclmaxlogfilesize`
Maximum Number of Log Files to Keep in Rotation	`orclmaxlogfiles`

To configure logging:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu, then select **Logging**.
2. Under Debug Level, select the types of activity to be logged.
3. Under Operations Enabled for Debug, enable the LDAP operations that you want logged.
4. Under Logging, specify values for Maximum log file size (MB) and Maximum number of log files to keep in rotation. The defaults are 1 MB and 100 log files, respectively.

> **Note:**
>
> Values you set on the Logging tab of the Server Properties page control LDAP server debugging.

24.3 Managing Logging from the Command Line

You can also manage the logging related tasks from the command line.

For more information, refer to the following sections:

- Viewing Log Files from the Command Line
- Setting Debug Logging Levels Using the Command Line

- Setting the Debug Operation Using the Command Line
- Force Flushing the Trace Information to a Log File

24.3.1 Viewing Log Files from the Command Line

You can view Oracle Internet Directory log files in a text editor.

See Table 24-1.

24.3.2 Setting Debug Logging Levels Using the Command Line

You set debug logging levels by using the `ldapmodify` command.

Because debug levels are additive, you must add the numbers representing the functions that you want to activate, and use the sum of those in the command-line option.

By default, debug logging is turned off. To turn it on, modify the attribute `orcldebugflag` in the instance-specific configuration entry to the level you want.

> **Note:**
>
> The DN of an instance-specific configuration entry has the form:
>
> `cn=componentname,cn=osdldapd, cn=subconfigsubentry`

You can configure debug levels to one of the following levels.

Table 24-3 shows values for OrclDebugFlag.

Table 24-3 Values for OrclDebugFlag

Value	Operation
1	Heavy trace debugging
128	Debug packet handling
256	Connection management
512	Search filter processing
1024	Entry parsing
2048	Configuration file processing
8192	Access control list processing
491520	Log of communication with DB
524288	Schema related operations
4194304	Replication specific ops
8388608	Log of entries, operations, and results for each connection
16777216	Trace function call arguments
67108864	Number and identity of clients connected to this server
117440511	All possible operations and data

Table 24-3 (Cont.) Values for OrclDebugFlag

Value	Operation
134217728	All Java plug-in debug messages and internal server messages related to the Java plug-in framework
268435456	All messages passed by a Java plug-in using the ServerLog object.
402653184	Both of the above

For example, to trace search filter processing (512) and connection management (256), enter 768 as the debug level (512 + 256 = 768).

You can use orcldebugflag to turn logging on and off. For example, to turn logging on by setting the value of orcldebugflag to 1 for the instance oid1, use this command:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f debugOn.ldif
```

where debugOn.ldif contains:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orcldebugflag
orcldebugflag: 1
```

To turn logging off, set the value of orcldebugflag to 0 for the instance. For example, to turn debugging off for the instance oid1, use this command:

```
ldapmodify -p oidPort -D cn=orcladmin -w adminPasswd -f debugOff.ldif
```

where debugOff.ldif contains:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orcldebugflag
orcldebugflag: 0
```

> **Note:**
>
> The value of orcldebugflag controls LDAP server debugging. To control Replication server debugging, set the value of orcldebuglevel, as described in Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify.

24.3.3 Setting the Debug Operation Using the Command Line

To make logging more focused by limiting logging to specific directory server operations, specify the debug operations you want logged using the orcldebugop attribute.

You can configure a subset of the values in Table 24-4 by adding the codes together. For example, to set debugging for ldapbind and ldapadd operations, set orcldebugop to 5 (1 + 4 = 5).

Table 24-4 Debug Operations for Setting the `orcldebugop` Attribute

Debug Operation	Provides Information Regarding
1	ldapbind
2	ldapunbind
4	ldapadd
8	ldapdelete
16	ldapmodify
32	ldapmodrdn
64	ldapcompare
128	ldapsearch
256	ldapabandon
511	All LDAP operations

To log more than one operation, add the values of their dimensions. For example, if you want to trace ldapbind (1), ldapadd (4) and ldapmodify (16) operations, then create an LDIF file setting the `orcldebugop` attribute to 21 (1 + 4 + 16 = 21). The LDIF file is as follows:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype:modify
replace:orcldebugop
orcldebugop:21
```

To load this file, enter:

```
ldapmodify -D "cn=orcladmin" -q -h host_name -p port_number -f file_name
```

24.3.4 Force Flushing the Trace Information to a Log File

To minimize the performance overhead in I/O operations, debug messages are flushed to the log file periodically instead of every time a message is logged by the directory server.

Writing to the log file is performed when one of the following occur:

- An LDAP operation completes
- A high priority message is logged
- The trace messages buffer is full

You can, however, view the trace messages in the log file as they are logged without having to wait for the periodic flush. To do this, set the instance-specific configuration entry attribute `orcldebugforceflush` to `1`. Do this by using ldapmodify as shown in the following example.

To enable force flushing by using `ldapmodify`:

1. Create an LDIF file as follows:

   ```
   dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
   changetype: modify
   replace: orcldebugforceflush
   orcldebugforceflush: 1
   ```

2. Load this file by entering the following:

   ```
   ldapmodify -D "cn=orcladmin" -q -h host_name -p port_number -f file_name
   ```

Note:

- When force flushing is enabled, the format of the trace message object for every operation becomes fragmented.
- By default, force flushing is disabled. After you have flushed the necessary information to the log file, you should disable force flushing.

See Also:

Oracle Identity Management LDAP Attribute Reference in *Reference for Oracle Identity Management* for information about the `orcldebugforceflush` attribute

25

Monitoring Oracle Internet Directory

This chapter describes the Oracle Internet Directory Manageability framework, which enables you to monitor statistics for Oracle Internet Directory. It describes how to configure and view statistics collection using Oracle Enterprise Manager Fusion Middleware Control, configure statistics using LDAP command-line utilities, and view statistics using Oracle Directory Services Manager (ODSM) and the `oiddiag` tool.

- Introduction to Monitoring Oracle Internet Directory Server
- Overview of Statistics Collection Using Fusion Middleware Control
- Overview of Statistics Information Viewable from Fusion Middleware Control
- Overview of Statistics Collection Using Metrics REST API
- Statistics Information Accessible from the Oracle Directory Services Manager Home Page
- Understanding Statistics Collection Using the Command-Line
- Viewing Information with the OIDDIAG Tool
- Monitoring Oracle Internet Directory Server Using LDAP

For information about monitoring other Oracle Fusion Middleware components, see Monitoring Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

25.1 Introduction to Monitoring Oracle Internet Directory Server

This section introduces you to how to monitor Oracle Internet Directory server.

For more information on how to monitor Oracle Internet Directory server, refer to the following sections:

- Capabilities of Oracle Internet Directory Server Manageability
- Oracle Internet Directory Server Manageability Architecture and Components
- Security Events and Statistics Entries Purge
- Account Used for Accessing Server Manageability Information

25.1.1 Capabilities of Oracle Internet Directory Server Manageability

The Oracle Internet Directory Server Manageability framework enables you to monitor the directory server statistics:

- Server health statistics about LDAP request queues, percent CPU usage, memory, LDAP sessions, and database sessions. For example, you can view the number of active database sessions over a period. You can also view the total number of connections opened to Oracle Internet Directory server instances over a period.
- Performance statistics. Average latency in millisecond is provided for bind, compare, messaging search, and all search operations over a period.

- General statistics about specific server operations, such as add, modify, or delete. For example, you can view the number of directory server operations over a period. You can also view the failed bind operation count.
- User statistics comprising successful and failed operations to the directory and the user performing each one. All LDAP operations are tracked for configured users. Also, the connections held by users at the ends of the statistics collection period are tracked.
- Critical events related to system resources and security—for example, occasions when a user provided the wrong password or had inadequate access rights to perform an operation. Other critical events include ORA errors other than expected errors including 1, 100 or 1403 and abnormal termination of the LDAP server.
- Security events tracking of users' successful and unsuccessful bind and userpassword compare operations.

  Because bind and user password compare are among the most security sensitive operations, an exclusive category security event is used to track these two operations. This event tracks the number of these operations performed by LDAP users and applications. The basic information recorded is user DN and source IP address. For failed user password compare, additional information is tracked, specifically, the number of failed compares of one user's password by another user from a given IP address.
- Status information of the directory server and the directory replication server—for example, the date and time at which the directory replication server was invoked

25.1.2 Oracle Internet Directory Server Manageability Architecture and Components

There are various relationships between various components of directory server manageability.

The relationship between the various components of directory server manageability is explained in Figure 25-1 and the accompanying text in Table 25-1.

Figure 25-1 Architecture of Oracle Internet Directory Server Manageability

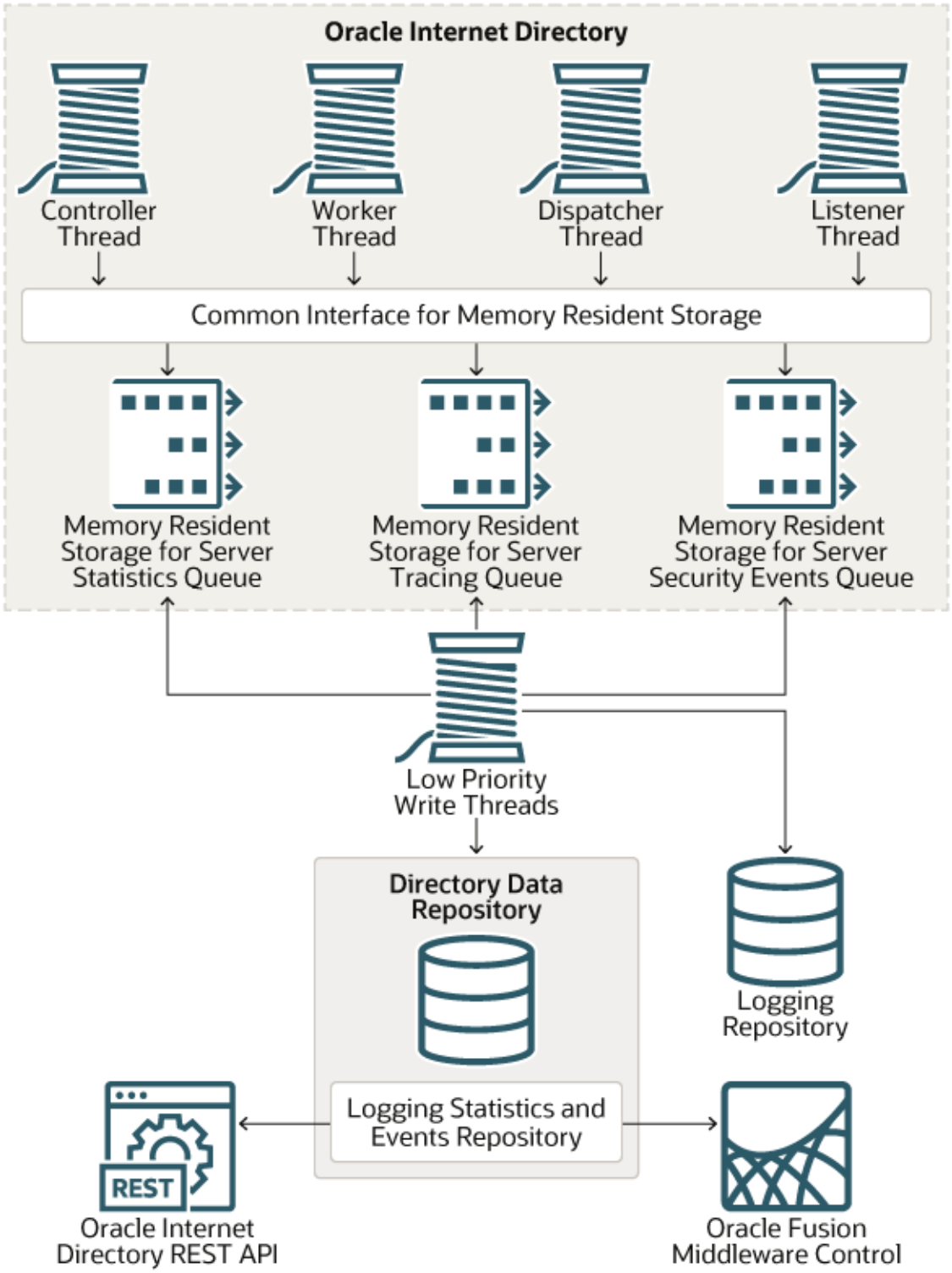


Table 25-1 Components of Oracle Internet Directory Server Manageability

Component	Description
Oracle Internet Directory	A directory server responds to directory requests from clients. It has four kinds of functional threads: controller, worker, dispatcher, and listener. It accepts LDAP requests from clients, processes them, and sends the LDAP response back to the clients. When you use the Oracle Internet Directory Server Manageability framework to set run-time monitoring, the four functional threads of the server record the specified information and store it in local memory. **See Also:** Oracle Directory Server Instance for a description of the directory server
Memory Resident Storage	This is a local process memory. The Oracle Internet Directory Server Manageability framework assigns one each for statistics, tracing, and security events. Each has its own separate data structure maintained in the local memory storage.
Low Priority Write Threads	These dedicated write threads differ from server functional threads in that they write server statistics, security events logging, and tracing information to the repository. To maintain reduced system overhead, their priorities are kept low.
External Monitoring Application	This module, which is proprietary and external to the server manageability framework, collects the gathered statistics through a standard LDAP interface with the directory server and stores it in its own repository.
External Repository for Server Management Information	This is the repository that the monitoring agent uses to store the gathered directory server statistics. The monitoring agent determines how this repository is implemented.
Fusion Middleware Control	extracts monitored data from the statistics and events repository, presenting it in a Web-based graphical user interface. Users can view the data in a normal browser. A repository can store the collected data for generic and custom queries.
Logging Repository (File System)	This repository uses a file system to store information traced across various modules of the directory server. By using a file system for this purpose, the Oracle Internet Directory Server Manageability framework uses the features and security of the operating system.
Directory Data Repository	This repository contains all user-entered data—for example, user and group entries.
Statistics and Events Repository	This repository is like the tracing repository except that it stores the information in the same database as the directory data repository rather than in a file system. In this way, the Oracle Internet Directory Server Manageability framework uses: • Normal LDAP operations to store and retrieve the information • Existing access control policies to manage the security of the gathered information The directory manageability framework isolates the gathered information from the directory data by storing the two separately.
Oracle Internet Directory Metrics REST API	Fetches monitored data from the statistics and events repository. Administrators can query this data using the REST API and use the data to present it in any desired dashboard to monitor Oracle Internet Directory.

25.1.3 Security Events and Statistics Entries Purge

Obsolete statistics entries are removed from Oracle Internet Directory by the Oracle Internet Directory purge tool.

Obsolete statistics entries are removed from Oracle Internet Directory by the Oracle Internet Directory purge tool, described in Managing Garbage Collection.

25.1.4 Account Used for Accessing Server Manageability Information

The Oracle Internet Directory database account `ODSSM` is used to access server manageability information from the database.

During installation, this account's password is set to a value provided by the user at a prompt. The credentials for this account, including the password, are stored in the Oracle Internet Directory snippet in the Oracle Enterprise Manager Fusion Middleware Control file `targets.xml`.

The only way you can change this account's password is to use the procedure documented in Changing the Password for the ODSSM Administrator Account. There is no support in the `oidpasswd` tool for changing this password.

25.2 Overview of Statistics Collection Using Fusion Middleware Control

You can view statistics using Fusion Middleware Control.

This section contains the following topics:

- Configuring Directory Server Statistics Collection Using Fusion Middleware Control
- Configuring a User for Statistics Collection Using Fusion Middleware Control

25.2.1 Configuring Directory Server Statistics Collection Using Fusion Middleware Control

You can configure directory server statistics collection by using Fusion Middleware control.

To configure statistics collection from Oracle Enterprise Manager Fusion Middleware Control, follow these steps:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu, then select **Statistics**.
2. In the General section of the page, select **Stats Flag** to enable statistics collection.
3. Specify the number of minutes in the Stats Frequency field to control the frequency of statistics collection.
4. Select values from the Bind Security Event Tracking and Compare Security Event Tracking lists.
5. To collect statistics about users, select **User Statistics Collection** in the User Statistics section of the page.
6. In the Event Levels section of the page, select the events you want to track.

Table 25-2 Configuration Attributes on Server Properties Page, Statistics Tab

Field or Heading	Configuration Attribute
Stats Flag	`orclstatsflag`
Stats Frequency (min)	`orclstatsperiodicity`
Bind Security Event Tracking and Compare Security Event Tracking	`orcloptracklevel`
User Statistics	`orclstatslevel`
Event Levels	`orcleventlevel`

> **Note:**
>
> - After you enable User Statistics collection, you also must specify individual users for statistics collection. See Configuring a User for Statistics Collection Using Fusion Middleware Control .
> - If you do not select SuperUser Login as an event level, the corresponding Security values on the Oracle Internet Directory home page is always 0.
> - The consecutive settings of `orcldebugflag` and of `orcloptracklevel` are additive.

25.2.2 Configuring a User for Statistics Collection Using Fusion Middleware Control

You can configure a user for statistics collection using Fusion Middleware control.

> **Note:**
>
> If you have configured `orclldapconntimeout` so that idle LDAP connections are closed after a period of time, as described in the LDAP Server Attributes in *Tuning Performance*, be aware that connections do not time out as per this setting for users who are configured for statistics collection.

To configure a user so that Server Manageability collects statistics for that user:

1. From the Oracle Internet Directory menu, select **Administration**, then **Shared Properties**.
2. Select the **General** tab.
3. Add the user's distinguished name to **User DN**. (This adds the user's DN to the attribute `orclstatsdn`.) For example:

   ```
   cn=Mary Lee, ou=Product Testing, c=uscn=Michael Smith, ou=Product Testing,
   c=uscn=Raj Sharma, ou=Human Resources, c=us
   ```

25.3 Overview of Statistics Information Viewable from Fusion Middleware Control

You can use Oracle Enterprise Manager Fusion Middleware Control to view many of the features of Oracle Internet Directory Server Manageability, as explained in this section.

For more information, refer to the following sections:

- Statistics Information Viewable from the Oracle Internet Directory Home Page
- Viewing Information on the Oracle Internet Directory Performance Page

25.3.1 Statistics Information Viewable from the Oracle Internet Directory Home Page

The Oracle Internet Directory home page displays Performance, Load, Security, Resource usage and Average response & Load related statistics information.

The Oracle Internet Directory Home Page displays the following information:

- Performance
  - Average Operation Response Time(ms)
  - Messaging Search Response Time(ms)
  - Bind Response Time(ms)
- Load
  - Total LDAP Connections
  - Operations Completed
  - Operations in progress
- Security
  - Failed Bind Operations
  - Failed SuperUser Logins
  - Successful SuperUser Logins
- Resource Usage
  - CPU Utilization%
  - Memory Utilization%
- Average Response and Load
  - LDAPserverResponse
  - numCompletedOps

Click Table View if you want to see values in tabular form.

In the Security section of the page, the values for Failed Bind Operations, Failed SuperUser Logins, and Successful SuperUser Logins are 0 if you have not enabled collection of these metrics. See Overview of Statistics Collection Using Fusion Middleware Control for more information.

25.3.2 Viewing Information on the Oracle Internet Directory Performance Page

The Oracle Internet Directory performance page displays performance summary information.

From the **Oracle Internet Directory** menu, select **Monitoring**, then **Performance Summary**. The following metrics are shown by default:

- Server Response
- Total Operations
- Messaging Search Operation Response Time

- Bind Operation Response Time
- Compare Operation Response Time
- Total Number of Security Events Objects in Purge Queue
- Total Number of Security Refresh Events Objects in Purge Queue
- Total Number of System Resource Events Objects in Purge Queue

To display other metrics, expand the Metrics Palette by clicking the arrow on the right edge of the window. You can collapse the Metrics Palette by clicking the arrow on the left edge of the window.

The default time interval is 15 minutes. To change the time interval, click **Slider**, then use the sliders to set the time interval. You can also click the **Date and Time** icon, set the start and end date and time on the Enter Date and Time dialog, then click **OK**.

Click the **Refresh** icon to refresh the page.

The **View** list enables you to view and save charts.

The **Overlay** list enables you to overlay the metrics for a different Oracle Internet Directory target.

> **Note:**
>
> - For non-critical events, there is a time lag of several minutes, up to `orclstatsperiodicity`, before the corresponding metric is updated.
> - You must click the Refresh icon to see updated metrics.

25.4 Overview of Statistics Collection Using Metrics REST API

You can use Oracle Internet Directory Metrics REST API to query OID runtime statistics information.

For more information, refer the following sections:

- Enabling Oracle Internet Directory Metrics REST API
- Securing Oracle Internet Directory Metrics REST API
- Statistics Information Accessible from Metrics REST API

25.4.1 Enabling Oracle Internet Directory Metrics REST API

You need to deploy Metrics REST API in the Weblogic server by extending the existing Weblogic Domain, where OID is installed. You need to use the **Oracle Internet Directory Metrics (Collocated)** Weblogic template to extend the existing Weblogic Domain. This will install the OID Metrics REST API along with a Weblogic JDBC data source.

25.4.2 Securing Oracle Internet Directory Metrics REST API

The REST API is protected by Basic Authentication mechanism in Weblogic. Users with "Administrators" or "OIDAdministrators" group membership have access to the metrics data.

25.4.3 Statistics Information Accessible from Metrics REST API

You can use Oracle Internet Directory Metrics REST API to query the OID runtime statistics information. Further, you can present the OID runtime statistics information in any desired dashboard application to monitor Oracle Internet Directory.

The following metrics information is available from the REST API for various operations:

- Performance
  - Average Operation Response Time (micro seconds)
  - Message Search Response Time (micro seconds)
  - Bind Response Time (micro seconds)
  - Compare Response Time (micro seconds)
- Load
  - Total LDAP Connections
  - Operations Completed
  - Operations in Progress
- Security
  - Failed Bind Operations

25.5 Statistics Information Accessible from the Oracle Directory Services Manager Home Page

You can access various statistics information from Oracle Directory Services Manager.

The Oracle Directory Services Manager home page for Oracle Internet Directory lists the following information:

- Uptime
- LDAP Connections
- OID Procs
- Number of Entries
- LDAP Change Log Entries
- Replication Agreements
- Debug Enabled
- Operation Latency

25.6 Understanding Statistics Collection Using the Command-Line

Using command-line utility, you can collect various statistics information as described in this section.

This section contains the following topics:

- Configuring Health, General, and Performance Statistics Attributes
- Configuring Security Events Tracking
- Configuring User Statistics Collection from the Command Line
- Configuring Event Levels from the Command Line
- Configuring a User for Statistics Collection Using the Command Line

25.6.1 Configuring Health, General, and Performance Statistics Attributes

You can use `ldapmodify` and `ldapsearch` to set and view statistics collection-related configuration attributes.

The attributes are in the instance-specific configuration entry, as described in Managing System Configuration Attributes.

To enable the collection of health, general, and performance statistics, set the `orclStatsFlag` and `orclStatsPeriodicity` attributes.

For example, to enable the Oracle Internet Directory Server Manageability framework for the component `oid1`, you create an LDIF file that looks like this:

```
dn:cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclstatsflag
orclstatsflag:1
```

To upload this file, enter the following command:

```
ldapmodify -h host -p port_number -D bind_DN -q -f file_name
```

where the bind DN authorized to perform server manageability configuration is `cn=emd admin,cn=oracle internet directory`.

25.6.2 Configuring Security Events Tracking

To configure security events tracking, set the attribute `orcloptracklevel`.

The attribute, `orcloptracklevel` is located in the instance-specific configuration entry, as described in Managing System Configuration Attributes. Table 25-3 lists the values of `orcloptracklevel` to configure different levels of bind and compare information collection:

Table 25-3 Values of orcloptracklevel

orcloptracklevel value	Configuration
1	Bind DN only
2	Bind DN and IP address
4	Compare DN only
8	Compare DN and IP address
16	Compare DN, IP address and failure details

The metrics recorded by each `orcloptracklevel` value are listed in the following table:

Table 25-4 Metrics Recorded by Each orcloptracklevel Value

Configuration	Metrics Recorded
DN only	Date and time stamp EID of DN performing the operation Success counts Failure counts
DN and IP address	All metrics listed under DN only Source IP Address
DN, IP address and failure details	All metrics listed under DN and IP address Distinct success counts Distinct failure counts Failure details for each DN performing password compare from an IP Address: • Date and time stamp • Source IP Address • EID of DN whose password is compared • Failure counts

The attributes `orcloptrackmaxtotalsize` and `orcloptracknumelemcontainers` enable you to tune memory used for tracking statistics and events.

See Tuning Security Event Tracking in *Tuning Performance*.

25.6.3 Configuring User Statistics Collection from the Command Line

To enable user statistics, set the `orclstatslevel` attribute to 1. The `orclStatsPeriodicity` attribute must also be set for user statistics collection to occur.

> **Note:**
>
> When you are collecting statistics for Oracle Enterprise Manager Fusion Middleware Control, set `orclStatsPeriodicity` to be the same as the collection periodicity of the Enterprise Manager agent, which is 10 minutes by default.

To configure users for statistics collection, see Configuring a User for Statistics Collection Using the Command Line.

25.6.4 Configuring Event Levels from the Command Line

The `orclstatsflag` attribute must be set to `1` for event level tracking to occur.

To configure event levels, use ldapmodify to set the `orcleventlevel` attribute to one or more of the event levels listed in Table 25-5. The attribute `orcleventlevel` is in the instance-specific configuration entry, as described in Managing System Configuration Attributes.

Table 25-5 Event Levels

Level Value	Critical Event	Information It Provides
1	SuperUser login	Super uses bind (successes or failures)
2	Proxy user login	Proxy user bind (failures)
4	Replication login	Replication bind (failures)
8	Add access	Add access violation
16	Delete access	Delete access violation
32	Write access	Write access violation
64	ORA 3113 error	Loss of connection to database
128	ORA 3114 error	Loss of connection to database
256	ORA 28 error	ORA-28 Error
512	ORA error	ORA errors other an expected 1, 100, or 1403
1024	Oracle Internet Directory server termination count	
2047	All critical events	

25.6.5 Configuring a User for Statistics Collection Using the Command Line

Using command line utility, you can configure a user for collecting statistics form the server.

> **Note:**
>
> If you have configured `orclldapconntimeout` so that idle LDAP connections are closed after a period of time, as described in the LDAP Server Attributes in *Tuning Performance*, be aware that connections do not time out as per this setting for users who are configured for statistics collection.

To configure a user by using the command line, add the user's DN to the DSA Configset entry's multivalued attribute `orclstatsdn` (DN: cn=dsaconfig,cn=configsets,cn=oracle internet directory) by using the `ldapmodify` command line tool. For example, this LDIF file adds Mary Lee to `orclstatsdn`:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype:modify
add: orclstatsdn
orclstatsdn: cn=Mary Lee, ou=Product Testing, c=us
```

Use a command line such as:

```
ldapmodify -h host -p port -f ldifFile -D cn=orcladmin -q
```

25.7 Viewing Information with the OIDDIAG Tool

Using OIDDIAG tool, you can view reports of various statistics.

Reports for all the statistics can be viewed using the `oiddiag` tool, as follows:

Security Events

```
oiddiag audit_report=true [outfile=file_name]
```

All Statistics and Events

```
oiddiag collect_all=true [outfile=file_name]
```

Subset of Statistics and Events

```
oiddiag collect_sub=true [infile=input_file_name outfile=file_name ]
```

where *input_file_name* is created by taking the output from

```
oiddiag listdiags=true
```

and removing unwanted statistics classes.

Statistics in HTML format

```
oiddiag collect_stats=true [outfile=file_name]
```

> **Note:**
>
> On Windows, the filename of the `oiddiag` command is `oiddiag.bat`.

> **Note:**
>
> Starting from this release, you can generate a HTML report which contains the following by supplying `collect_stats=true` argument:
>
> - Instance Statistics
> - Operations Statistics
> - Memory/CPU Usage Statistics
> - Network Bytes Sent/Received
> - Client Connections/Operations Statistics
> - DB Connections Statistics
> - LDAP Connections Statistics
> - Replication Operations Statistics
> - Replication Queue Statistics (for all replication agreements)

> **See Also:**
>
> - Oracle Internet Directory Server Diagnostic Command-Line Tool in *Reference for Oracle Identity Management* for information about `oiddiag` command tool
> - The chapter about Overview of Oracle Fusion Middleware Administration Tools in *Administering Oracle Fusion Middleware*

25.8 Monitoring Oracle Internet Directory Server Using LDAP

Oracle Internet Directory provides a variety of methods to monitor the current state of the server for debugging or troubleshooting purposes.

You can monitor the server over LDAP in the following ways:

- Viewing Monitoring Information Using the cn=monitor Entry
- Viewing the Available Monitoring Information
- Monitoring System Information
- Monitoring the Work Queue
- Monitoring Client Connections
- Monitoring Version Information

25.8.1 Viewing Monitoring Information Using the cn=monitor Entry

Oracle Internet Directory 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.

The first level entries under `cn=monitor` are the categories of available monitoring information. For each category, there is only one attribute called `orclmetricsummary`, which includes all the monitoring information for that category in the report form. Subtypes of `orclmetricsummary` further describe the monitoring information.

> **Note:**
>
> Monitoring information using the `cn=monitor` entry is enabled by default. However, care must be taken to avoid overusing this capability as it can adversely affect performance.

25.8.2 Viewing the Available Monitoring Information

Use the `ldapsearch` command with the base DN `"cn=monitor"` to print the supported categories of monitoring information.

```
$ ldapsearch -h localhost -p 3060 -D cn=orcladmin -w password -b "cn=monitor"
"objectclass=*" -s base
```

The output will be similar to the following:

```
cn=monitor
orclmetricsummary;categories=cn=system information,cn=monitor
cn=work queue,cn=monitor
cn=client connections,cn=monitor
cn=version,cn=monitor
```

25.8.3 Monitoring System Information

Use the `ldapsearch` command with the base DN `"cn=System Information,cn=monitor"` to display general system information like OS version.

```
$ ldapsearch -h localhost -p 3060 -D cn=orcladmin -w password -b "cn=system
information,cn=monitor" "objectclass=*" -s base
```

The output will be similar to the following:

```
cn=system information,cn=monitor
orclmetricsummary;os_version=Linux 5.15.0-208.159.3.el8uek.x86_64 x86_64
```

25.8.4 Monitoring the Work Queue

Use the `ldapsearch` command with base DN `"cn=work queue,cn=monitor"` to keep track of outstanding client requests and ensures that they are processed. It displays work queue size and latency information.

```
$ ldapsearch -h localhost -p 3070 -D cn=orcladmin -w password -b "cn=work
queue,cn=monitor" "objectclass=*" -s base
```

The output will be similar to the following:

```
cn=work queue,cn=monitor
orclmetricsummary;overall_stats=total_queue_size:0,
avg_queue_latency:3258.000000 micro sec
orclmetricsummary;/inst1/oid1/1=total_queue_size:0, avg_queue_latency:3042
micro sec
orclmetricsummary;/inst1/oid2/1=total_queue_size:0, avg_queue_latency:3474
micro sec
```

25.8.5 Monitoring Client Connections

Use the `ldapsearch` command with base DN `"cn=client connections,cn=monitor"` to display the connection throughput information per instance and the top performing connections details.

```
$ ldapsearch -h localhost -p 3060 -D cn=orcladmin -w password -b "cn=client
connections, cn=monitor" "objectclass=*" -s base
```

The output will be similar to the following:

```
cn=client connections,cn=monitor
orclmetricsummary;/inst1/oid1/1=open_connections:0, new_connections:21,
closed_connections:21
orclmetricsummary;/inst1/oid2/1=open_connections:0, new_connections:20,
closed_connections:20
orclmetricsummary;top_connections=user_dn:cn=orcladmin,
user_ip:::ffff:10.196.16.156, open_connections:0, total_ops:43, bind_ops:21,
```

```
proxy_bind_ops:0, unbind_ops:21, compare_ops:0, search_base_ops:1,
search_one_ops:0, search_sub_ops:0, add_ops:0, delete_ops:0, modify_ops:0,
modrdn_ops:0, abandon_ops:0
user_dn:cn=emd admin,cn=oracle internet directory,
user_ip:::ffff:10.196.16.156, open_connections:0, total_ops:40, bind_ops:20,
proxy_bind_ops:0, unbind_ops:20, compare_ops:0, search_base_ops:0,
search_one_ops:0, search_sub_ops:0, add_ops:0, delete_ops:0, modify_ops:0,
modrdn_ops:0, abandon_ops:0
```

> **Note:**
>
> OID can only identify the IP addresses that it receives at its end. If you are using a load balancer then OID cannot identify the actual IP address of the client.

25.8.6 Monitoring Version Information

Use the `ldapsearch` command with base DN `"cn=version,cn=monitor"` to display Oracle Internet Directory and database version.

```
$ ldapsearch -h localhost -p 3070 -D cn=orcladmin -w password -b "cn=version,
cn=monitor" "objectclass=*" -s base
```

The output will be similar to the following:

```
cn=version,cn=monitor
orclmetricsummary;oid_version=14.1.2.1.0
orclmetricsummary;db_version=23.0.0.0.0
```

26

Backing Up and Restoring Oracle Internet Directory

All of the Oracle Internet Directory data that you need to back up and restore is in the underlying Oracle Database. You can back up a small directory or a specific naming context by using the `ldifwrite` utility.
For more information about backing up and restoring an Oracle Database, see the *Oracle Database Backup and Recovery User's Guide*.

To back up and restore your Oracle home and Oracle instance, see Introduction to Backup and Recovery in *Administering Oracle Fusion Middleware*.

> **Note:**
>
> If you back up data from an earlier version of Oracle Internet Directory, such as 10*g* Release 2 (10.1.2.0.2), then restore it on a node running 10*g* (10.1.4.0.1) or later, you must update the password policy entries.

27

Managing Quality of Service Configuration

You can manage the quality of service policies by configuring the `orclqosconfig` attribute using the LDAP command-line tools.

This chapter includes the following topics:

- Understanding Quality of Service in Oracle Internet Directory
- Configuring QoS to Restrict the Number of User Operations
- Restricting the Number of Connections from one DN
- Restricting the Number of Operations per Minute for a DN
- Restricting the Number of Add Operations for a User
- Restricting All Operations Using `allop`
- Restricting the Number of Operations on a DSA Config Entry

27.1 Understanding Quality of Service in Oracle Internet Directory

Oracle Internet Directory provides the capability to restrict the number of user connections or operations possible in a given duration of time.

This helps Oracle Internet Directory in maintaining its stability and performance. This is done on per DN basis and must be valid per instance basis.

Oracle Internet Directory provides the following capabilities:

- Restrict the number of connections from one DN
- Restrict the number of operations per minute for a DN
- Restrict the number of operations on a DSA config entry

The configuration for this Quality of Service capability is stored in OID DSA config entry in a new multi-valued attribute `orclqosconfig`. The `orclqosconfig` attribute is a configuration attribute in the DSA Configuration Entry: `cn=dsaconfig,cn=configsets,cn=oracle internet directory`.

`orclqosconfig` is a multivalued attribute, so multiple attributes and their respective values can be computed dynamically.

27.2 Configuring QoS to Restrict the Number of User Operations

You can configure the `orclqosconfig` attribute to restrict the number of user connections or operations for a duration of time.

The syntax of `orclqosconfig` is:

```
orclqosconfig;ENTRYDN:numofconns:Value,add:Value,del:Value,mod:Value,modrdn:Value,search:Value,compare:Value,allop:Value,duration:Value
```

where,

- `ENTRYDN`: Entry DN on which throttling has to be done.
- `numofconns,Value`: Total number of simultaneous connections allowed for ENTRYDN. The number of connections must be greater than 0 (`numofconns` >0). It does not accept a negative or zero value.
- `add,Value`: Total number of add operations allowed per minute.
- `del,Value`: Total number of delete operations allowed per minute.
- `mod,Value`: Total number of modify operations allowed per minute.
- `modrdn,Value`: Total number of `modrdn` (or `moddn`) operations allowed per minute. The `modrdn` parameter includes both `moddn` and `modrdn` operations.
- `search,Value`: Total number of search operations allowed per minute.
- `compare,Value`: Total number of compare operations allowed per minute.
- `allop,Value`: A common configuration value for all the above operations. Instead of setting operation for each value, here you can set it once and it will be used for all.
- `duration:Value`: Time period in seconds to restrict the number of user connections or operations. The default is 60 seconds.

> **Note:**
>
> - All the above parameters are optional and duplicity of entry DN in configuration value is not allowed.
> - The valid values must be greater than 0.

27.3 Restricting the Number of Connections from one DN

You can control the number of connections a particular user or users can make from a single DN. This restricts some heavy usage clients, which create/delete tenants and ensures that overall experience with OID remains good for other clients.

To restrict the number of connections from a single DN, run the following command:

```
$ ldapmodify -h localhost -p port -D cn=orcladmin -q -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype:modify
replace: orclqosconfig
orclqosconfig;cn=admin,ou=hr,ou=americas,o=IMC,c=US:numofconns:3,duration:60
```

27.4 Restricting the Number of Operations per Minute for a DN

You can restrict the number of operations allowed per minute for a given DN.

To restrict the number of operations, run the following command:

```
$ ldapmodify -h localhost -p port -D cn=orcladmin -q -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype:modify
replace: orclqosconfig
orclqosconfig;cn=admin,ou=hr,ou=americas,o=IMC,c=US:del:3,mod:2,duration:60
```

27.5 Restricting the Number of Add Operations for a User

You can restrict the number of add operations with default duration for a user.

To restrict the number of add operations with default duration, run the following command:

```
$ ldapmodify -h localhost -p port -D cn=orcladmin -q -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype:modify
replace: orclqosconfig
orclqosconfig;cn=admin,ou=hr,ou=americas,o=IMC,c=US:add:5,duration:60
```

27.6 Restricting All Operations Using `allop`

You can restrict all operations for a given DN within a minute.

To restrict all operations, run the following command:

```
$ ldapmodify -h localhost -p port -D cn=orcladmin -q -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype:modify
replace: orclqosconfig
orclqosconfig;cn=admin,ou=hr,ou=americas,o=IMC,c=US: allop:2
```

27.7 Restricting the Number of Operations on a DSA Config Entry

You can restrict the number of modifications on a DSA config entry per minute by configuring the attribute `orclqosconfig;dsaConfig:Value`. `Value` denotes the number of modifies allowed per minute on a DSA config entry.

To restrict the number of modifications on a DSA config entry, run the following command:

```
$ ldapmodify -h localhost -p port -D cn=orcladmin -q -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype:modify
replace: orclqosconfig
orclqosconfig;dsaConfig:mod:5,duration:60
```

Part III

Advanced Administration: Security

In Oracle Internet Directory, you can secure data, establish access controls, manage password policies and store various data, using administration module.

This part explains how to:

- Secure data within the directory
- Establish access controls for administering applications in enterprises and hosted environments
- Establish and manage policies governing passwords
- Manage password verifiers used to authenticate users to other Oracle components
- Store data for users, groups, and services in one repository, and delegate the administration of that data to various administrators

It contains these chapters:

- Configuring Secure Sockets Layer (SSL)
- Configuring Data Privacy
- Managing Password Policies
- Managing Directory Access Control
- Managing Password Verifiers
- Delegating Privileges for Oracle Identity Management
- Managing Authentication

28

Configuring Secure Sockets Layer (SSL)

This chapter describes how to configure Secure Sockets Layer (SSL) for use with using Oracle Enterprise Manager Fusion Middleware Control, and LDAP command-line utilities. It also describes how to test an SSL connection using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities and how to set the SSL interoperability mode for compatibility with Oracle components developed before 11g Release 1 (11.1.1.0.0).
If you use SSL for Oracle Internet Directory, you might also want to configure strong authentication, data integrity, and data privacy.

This chapter includes the following sections:

- Overview of Configuring Secure Sockets Layer (SSL)
- Overview of Configuring SSL by Using Fusion Middleware Control
- Overview of Configuring SSL by Using LDAP Commands
- Configuring SSL in Oracle Internet Directory
- Configuring ODSM connection with SSL enabled
- Testing SSL Connections by Using Oracle Directory Services Manager
- Overview of Testing SSL Connections From the Command Line
- Configuring SSL between Database and Oracle Internet Directory

See Also:

- Security Features in Oracle Internet Directory for a conceptual overview of SSL in relation to Oracle Internet Directory.
- Managing Authentication
- See Configuring Secure Sockets Layer (SSL) in *Administering Oracle Fusion Middleware*
- See SSL Automation Tool in *Administering Oracle Fusion Middleware*. The SSL Automation Tool enables you to configure SSL for multiple components using a domain-specific CA.

28.1 Overview of Configuring Secure Sockets Layer (SSL)

Oracle Internet Directory ensures that data has not been modified, deleted, or replayed during transmission by using Secure Sockets Layer (SSL). SSL generates a cryptographically secure message digest—through cryptographic checksums —and includes it with each packet sent across the network. SSL provides authentication, encryption, and data integrity using message digest.

This introduction contains the following topics:

- Supported Cipher Suites

- Supported Protocol Versions
- About SSL Authentication Modes
- Other Components and SSL
- SSL Interoperability Mode
- StartTLS

Oracle Internet Directory ensures that data is not disclosed during transmission by using public key encryption available with SSL. In public-key encryption, the sender of a message encrypts the message with the public key of the recipient. Upon delivery, the recipient decrypts the message using the recipient's private key.

28.1.1 Supported Cipher Suites

A cipher suite is a set of authentication, encryption, and data integrity algorithms used for exchanging messages between network nodes. During an SSL handshake, the two nodes negotiate to determine which cipher suite they will use when transmitting messages back and forth.

The following table lists the TLS cipher suites supported by Oracle Internet Directory and their corresponding authentication, encryption, and data integrity mechanisms. These are stored in the attribute `orclsslciphersuite` in the instance-specific configuration entry.

> **Note:**
>
> Ensure that you have Java Development Kit (JDK) 17 or higher installed on your system.

Table 28-1 TLS Cipher Suites Supported in Oracle Internet Directory

Cipher Suite	Authentication	Encryption	Data Integrity	TLS Compatibility
TLS_AES_128_CCM_SHA256	ECDHE_RSA, DHE_RSA, ECDHE_ECDSA	AES 128 CCM	SHA256 (SHA 2)	TLS 1.3
TLS_AES_128_GCM_SHA256	ECDHE_RSA, DHE_RSA, ECDHE_ECDSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.3
TLS_AES_256_GCM_SHA384	ECDHE_RSA, DHE_RSA, ECDHE_ECDSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.3
TLS_CHACHA20_POLY1305_SHA256 (non-FIPS only)	ECDHE_RSA, DHE_RSA, ECDHE_ECDSA	CHACHA20/ POLY1305	SHA256 (SHA-2)	TLS 1.3
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256	DHE_RSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384	DHE_RSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	ECDHE_ECDSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2

Table 28-1 (Cont.) TLS Cipher Suites Supported in Oracle Internet Directory

Cipher Suite	Authentication	Encryption	Data Integrity	TLS Compatibility
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	ECDHE_ECDSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDHE_RSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	ECDHE_RSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2

The following table lists the less secure TLS cipher suites supported by Oracle Internet Directory for legacy products, along with their corresponding authentication, encryption, and data integrity mechanisms.

Table 28-2 TLS Cipher Suites Supported in Oracle Internet Directory for Legacy Products

Cipher Suite	Authentication	Encryption	Data Integrity	TLS Compatibility
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256	DHE_RSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	DHE_RSA	AES 256 CBC	SHA (SHA-1)	TLS 1.2
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256	DHE_RSA	AES 256 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	ECDHE_ECDSA	AES 128 CBC	SHA (SHA-1)	TLS 1.2
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	ECDHE_ECDSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	ECDHE_ECDSA	AES 256 CBC	SHA (SHA-1)	TLS 1.2
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	ECDHE_ECDSA	AES 256 CBC	SHA384 (SHA-2)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDHE_RSA	AES 128 CBC	SHA (SHA-1)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDHE_RSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	ECDHE_RSA	AES 256 CBC	SHA (SHA-1)	TLS 1.2
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	ECDHE_RSA	AES 256 CBC	SHA384 (SHA-2)	TLS 1.2

Table 28-2 (Cont.) TLS Cipher Suites Supported in Oracle Internet Directory for Legacy Products

Cipher Suite	Authentication	Encryption	Data Integrity	TLS Compatibility
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	AES 128 CBC	SHA (SHA-1)	TLS 1.2
TLS_RSA_WITH_AES_128_CBC_SHA256	RSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_RSA_WITH_AES_128_GCM_SHA256	RSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	AES 256 CBC	SHA (SHA-1)	TLS 1.2
TLS_RSA_WITH_AES_256_CBC_SHA256	RSA	AES 256 CBC	SHA256 (SHA-2)	TLS 1.2
TLS_RSA_WITH_AES_256_GCM_SHA384	RSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2

Note:

Anonymous ciphers are not enabled by default. If there is any requirement, you can add the `TLS_DH_anon_WITH_AES_256_GCM_SHA384`, `TLS_DH_anon_WITH_AES_128_GCM_SHA256` and `SSL_DH_anon_WITH_3DES_EDE_CBC_SHA` cipher suites into Oracle Internet Directory configuration using the Oracle Fusion Middleware System MBean Browser.

28.1.2 Supported Protocol Versions

The list of all the enabled protocols along with their attribute values are mapped in this topic.

Note:

The out-of-box default SSL configuration of OID server instances has the value of `orclcryptoversion` set to 48. This means, TLSv1.2 and TLSv1.3 are enabled by default.

Oracle Internet Directory supports the following TLS protocols:

- TLSv1.2
- TLSv1.3

> **Note:**
>
> Oracle Internet Directory does not support SSLv2, SSLv3, TLSv1.0, and TLSv1.1.

You can specify the TLS version using the `orclcryptoversion` attribute.

The `orclcryptoversion` attribute allows you to enable more than one protocol by specifying the corresponding value and populating the attribute.

Table 28-3 lists the protocol mapping with its corresponding value.

You can completely disable TLS 1.2 by updating the value of `orclcryptoversion` to `32` (value of TLS 1.3 in the Protocol Mapping table).

To modify the value of `orclcryptoversion` to 32, use `ldapmodify` as follows:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

where ldifFile contains:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclcryptoversion
orclcryptoversion: 32
```

Table 28-3 Protocol Mapping

Enabled Protocol	Attribute Value
All Supported Protocols	0
TLS 1.2	16
TLS 1.3	32
TLS 1.2 or TLS 1.3	48

28.1.3 About SSL Authentication Modes

The SSL protocol provides transport layer security with authenticity, integrity, and confidentiality, for a connection between a client and server.

Three authentication modes are supported, as described in Table 28-4. The SSL authentication mode is controlled by the attribute `orclsslauthentication` in the instance-specific configuration entry.

By default, Oracle Internet Directory uses SSL No Authentication Mode (`orclsslauthentication=1`).

When both a client and server authenticate themselves with each other, SSL derives the identity information it requires from the X509v3 digital certificates.

> **See Also:**
>
> Managing Authentication .

> **Note:**
>
> By default, the SSL authentication mode is set to 1 (encryption only, no authentication).
>
> If you are using Oracle Delegated Administration Services 10g or other client applications such as legacy versions of Oracle Forms and Oracle Reports that expect to communicate with Oracle Internet Directory on an encrypted SSL port configured for anonymous SSL ciphers, then at least one Oracle Internet Directory server instance must be configured for this default authentication mode.
>
> Otherwise, authentication mode 1 and anonymous SSL ciphers are not required for Oracle Internet Directory to function. The type of SSL ports that are made available and the ciphers that the SSL port will accept depend on your specific deployment requirements.

During start-up of a directory server instance, the directory reads a set of configuration parameters, including the parameters for the SSL profile.

To run a server instance in secure mode, configure a single listening endpoint to communicate using LDAPS. To allow the same instance to run non-secure connections concurrently, configure a second listening endpoint to communicate using LDAP.

During installation of Oracle Internet Directory, Oracle Identity Management 14*c* Installer follows specific steps in assigning the SSL and non-SSL port. First, it attempts to use 3060 as the non-SSL port. If that port is unavailable, it tries ports in the range 3061 to 3070, then 13060 to 13070. Similarly, it attempts to use 3131 as its SSL port, then ports in the range 3132 to 3141, then 13131 to 13141.

> **Note:**
>
> If you perform an upgrade from an earlier version of Oracle Internet Directory to the current release, your port numbers from the earlier version are retained.

You can create and modify multiple Oracle Internet Directory instances with differing values, using a different SSL parameters. This is a useful way to accommodate clients with different security needs.

> **See Also:**
>
> Managing Oracle Internet Directory Instances for information about creating a new server instance.

28.1.4 SSL Authentication Modes

There are three modes of SSL authentication supported: SSL No Authentication Mode, SSL Server Authentication Only Mode and SSL Client and Server Authentication Mode.

Table 28-4 lists the authentication methods, its values and behavior.

Table 28-4 SSL Authentication Modes

SSL Authentication Method	Value of orclsslauthentication	Authentication Behavior
SSL No Authentication Mode, Confidentiality mode	1	Neither the client nor the server authenticates itself to the other. No certificates are sent or exchanged. Only SSL encryption and decryption is used. **Note:** If Oracle Internet Directory SSL is configured in SSL No Authentication Mode then you must enable the anonymous ciphers for Java clients to communicate with Oracle Internet Directory.
SSL Server Authentication Only Mode	32	The directory server authenticates itself to the client. The directory server sends the client a certificate asserting the server's identity.
SSL Client and Server Authentication Mode	64	The client and server authenticate themselves with each other and send certificates to each other.

Note:

If Java Development Kit (JDK) 1.8.0_201 or higher is installed on your system then the anonymous ciphers are disabled by default.

If Oracle Internet Directory SSL is configured in SSL No Authentication Mode then you must enable the anonymous ciphers in the JDK by editing the `java.security` file (`JAVA_HOME/lib/security`) and removing `anon` and `NULL` from the `jdk.tls.disabledAlgorithms` security property.

28.1.5 Oracle Wallets

Oracle Wallet is a secure software container that is used to store X509 certificates, Private key, and trusted CA certificates A self-signed certificate can be stored in Oracle Wallet that can be within an enterprise.

Before removing the reference to the wallet from the instance-specific configuration, you must disable SSL by setting `orclsslenable` to 0.

See Also:

Keystores and Oracle Wallets in *Administering Oracle Fusion Middleware* for information on using Oracle wallets with middleware components.

Never delete a wallet currently in use, as defined in the attribute `orclsslwalleturl`, from the file system. Doing so prevents the server from starting successfully. Remove the reference to the wallet from the instance-specific configuration entry attribute `orclsslwalleturl` before you delete the file.

You do not need to directly manipulate `orclsslwalleturl` because the SSL configuration service abstracts this out, both in WLST and Oracle Enterprise Manager Fusion Middleware Control. The SSL configuration service traps any attempts to delete a wallet that is currently in use, provided you do so by using the SSL configuration service.

28.1.6 Other Components and SSL

At installation, Oracle Internet Directory starts up in dual mode. That is, some components can access Oracle Internet Directory using non-SSL connections, while others use SSL when connecting to the directory.

By default, Oracle Application Server components are configured to run in this dual mode environment when communicating with Oracle Internet Directory. If you want, you can remove the non-SSL mode and change all middleware instances to use SSL.

Enterprise User Security or a customer application might need an SSL channel with a different configuration from the default. For example, it might need SSL server authentication mode or SSL mutual authentication mode. In this case, you must create another Oracle Internet Directory component instance listening on a different SSL mode and port.

> **See Also:**
>
> Managing Oracle Internet Directory Instances for instructions on how to configure server instances

For more information about Enterprise User Security SSL configuration, please see the section on enterprise user security configuration in *Oracle Database Enterprise User Administrator's Guide*.

28.1.7 SSL Interoperability Mode

In no-auth mode, Oracle legacy components developed before 11g Release 1 (11.1.1.0.0) such as legacy LDAP C clients can connect with Oracle Internet Directory only by using an instance that has interoperability mode enabled (`orclsslinteropmode = 1`).

The default value for SSL interoperability mode is now disabled (`orclsslinteropmode = 0`), in order to be fully compliant with the JDK SSL.

New clients using JSSE (Java Secure Socket Extensions) and non-Oracle clients need an SSL instance with the interoperability mode disabled. Oracle Internet Directory is fully compliant with the Sun JDK's SSL, provided SSL interoperability mode is disabled (`orclsslinteropmode = 0`).

If Oracle Internet Directory is set to the wrong mode for a client, you might observe rare and non-deterministic failures of client SSL connections to the server.

28.1.8 StartTLS

The Oracle Internet Directory supports startTLS. This feature enables the on-demand negotiation of an SSL session on a non-SSL port. No special configuration is required for the non-SSL port.

If Oracle Internet Directory has an SSL endpoint configured, a client can use startTLS on the non-SSL port to negotiate an SSL connection on the non-SSL port with the same configuration that is on the SSL port. That is, if the SSL port uses mutual authentication, startTLS tries to negotiate mutual authentication on the non-SSL port.

28.2 Overview of Configuring SSL by Using Fusion Middleware Control

You can configure SSL using Fusion Middleware Control by creating a wallet, configuring SSL parameters, and by setting the SSL parameters.

This section includes the following topics:

- Configuring SSL by Using Fusion Middleware Control
- Creating a Wallet by Using Fusion Middleware Control
- Configuring SSL Parameters by Using Fusion Middleware Control
- SSL Parameters with Fusion Middleware Control

28.2.1 Configuring SSL by Using Fusion Middleware Control

Configuring SSL by using Fusion Middleware Control consists of three basic tasks:

Follow the below given steps:

1. Creating a Wallet by Using Fusion Middleware Control
2. Configuring SSL Parameters by Using Fusion Middleware Control
3. Restarting Oracle Internet Directory.

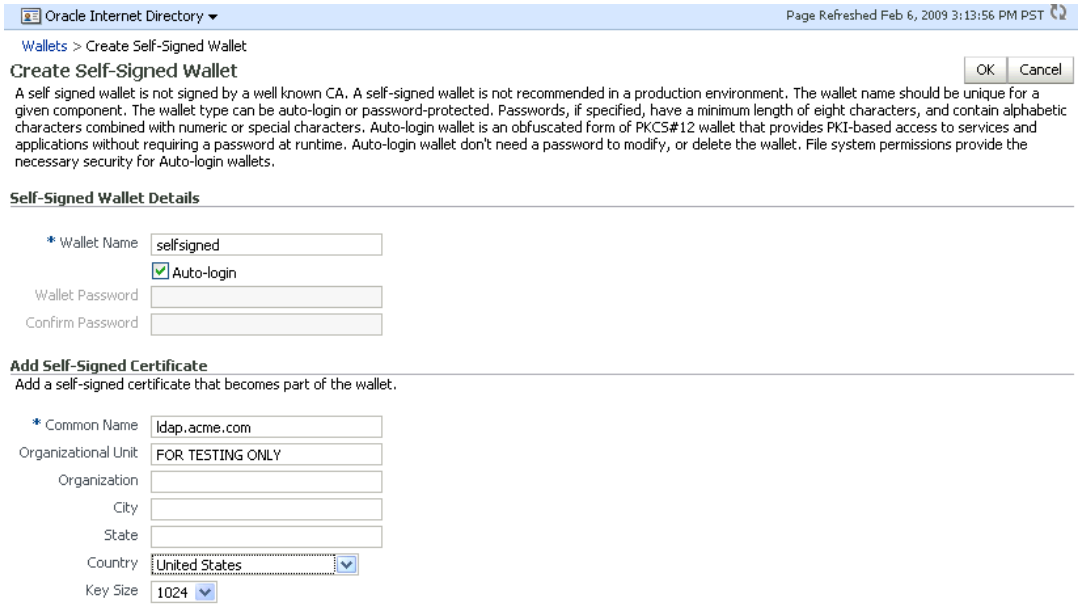
> **See Also:**
>
> - Managing Oracle Internet Directory Instances for instructions on how to stop and start the server.
> - "About Certificate Authentication Method by Using Fusion Middleware Control "
> - Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management* for descriptions of the SSL parameters

28.2.2 Creating a Wallet by Using Fusion Middleware Control

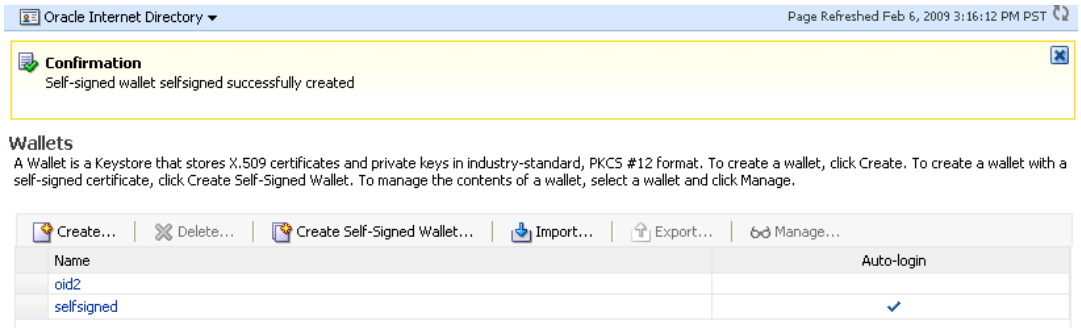
You can create a self-signed wallet to use when configuring SSL.

Perform the following steps:

1. From the Oracle Internet Directory menu, select **Security**, then **Wallets**. If any wallets exist, you see a list.
2. To create a new wallet, click **Create Self-signed Wallet**. The Create Self-Signed Wallet page appears.



3. On the Create Self-Signed Wallet page, enter a name for the new wallet, using lower-case letters only.
4. Select **Auto Login** for an auto login wallet. Wallets configured for Oracle Internet Directory must have auto login enabled.
5. If you have deselected **Auto Login**, enter the password in the two fields.
6. For **Common Name**, enter the hostname of the instance.
7. Select a **Key Size** from the list.
8. Click **Submit**.
9. A confirmation message is displayed and the new wallet appears in the list of wallets.



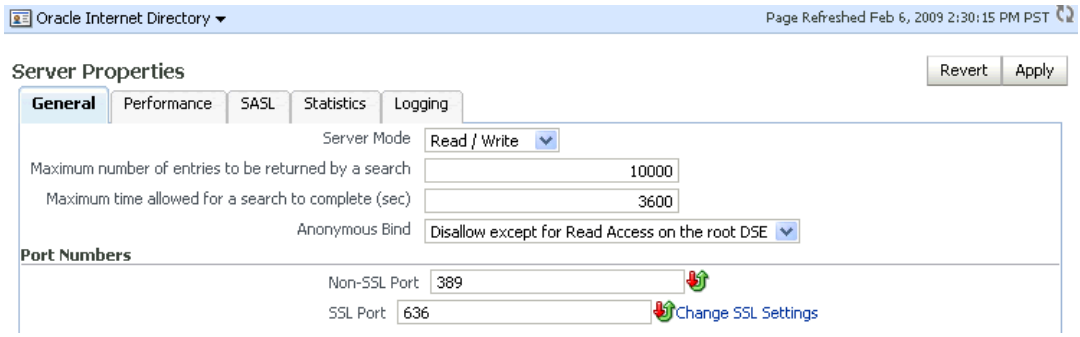
> **See Also:**
>
> Managing Keystores, Wallets, and Certificates in *Administering Oracle Fusion Middleware* for more information about Oracle wallets.

28.2.3 Configuring SSL Parameters by Using Fusion Middleware Control

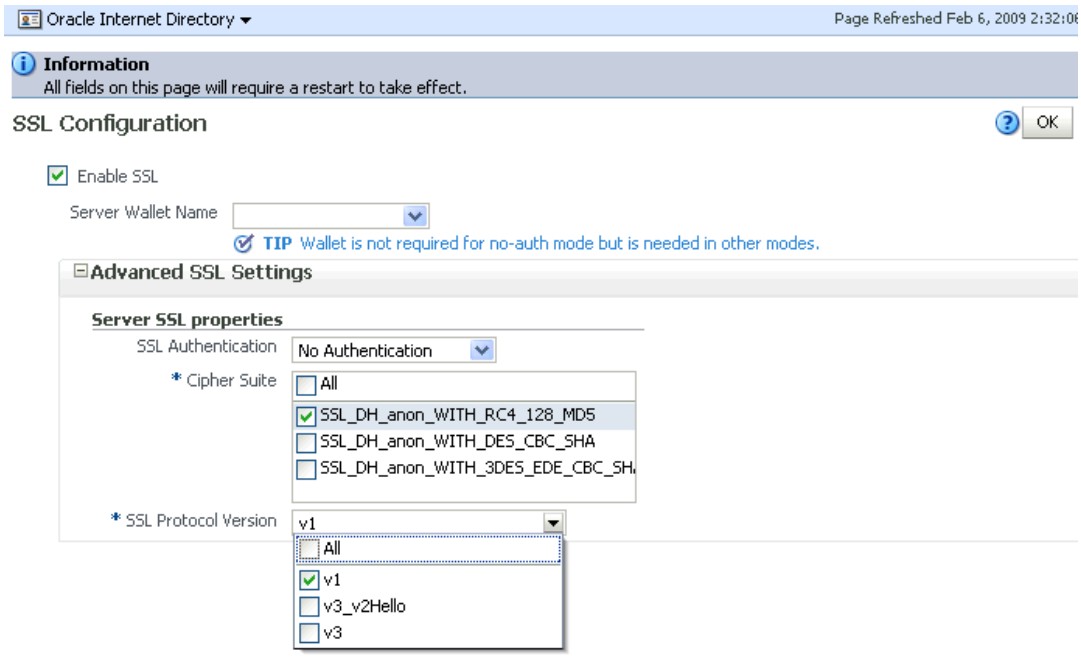
You can use Fusion Middleware control to configure SSL parameters.

After you have a wallet to use for configuring SSL, perform the following steps:

1. From the Oracle Internet Directory menu, select **Administration**, then **Server Properties**.



2. Click **Change SSL Settings**.
3. On the SSL Settings dialog:



- Select **Enable SSL**.
- Select a wallet.
- If this is a non auto login wallet, supply the wallet password in the Server Wallet Password field.
- If necessary, expand **Advanced SSL Settings**.
- Set SSL Authentication to **Server**.

- Set Cipher Suite to **All**.
- Set SSL protocol version to the appropriate version, usually **v3**.
- Click **OK**.

> **Note:**
>
> Currently, there is no way to configure TLS v1.1 and TLS v1.2 protocols and their corresponding ciphers from the OID Enterprise Manager Fusion Middleware Control. As a workaround, you can use the `ldapmodify` command to change the `orclcryptoversion` attribute. See Supported Protocol Versions.

4. Restart the Oracle Internet Directory instance by navigating to **Oracle Internet Directory**, then **Availability**, then **Restart.**

The steps for SSL-enabling in mutual-auth mode are the same, except that in the SSL Settings dialog, you would set SSL Authentication to **Mutual** instead of **Server**.

> **Note:**
>
> You cannot directly change the parameters for an active instance.

28.2.4 SSL Parameters with Fusion Middleware Control

This section lists the SSL parameters in Oracle Enterprise Manager Fusion Middleware Control that are applicable to Oracle Internet Directory.

Table 28-5 lists the SSL parameters in Oracle Enterprise Manager Fusion Middleware Control that are applicable to Oracle Internet Directory. All of them are in the instance-specific configuration entry, which has a DN of the form:

`"cn=componentname,cn=osdldapd,cn=subconfigsubentry."`

> **Note:**
>
> While setting up TLS in Oracle Internet Directory, Do not change the value for `orclsslversion` and retain its default value of 3.

Table 28-5 SSL-Related Attributes in Fusion Middleware Control

Field or Heading	Configuration Attribute
Server SSL Protocol Version	`orclsslversion`
SSL Wallet URL	`orclsslwalleturl`
Enable SSL	`orclsslenable`
SSL Authentication Mode	`orclsslauthentication`
Server Cipher Suite	`orclsslciphersuite`

You must restart the server for SSL configuration changes to take effect.

28.3 Overview of Configuring SSL by Using LDAP Commands

You can configure SSL using LDAP commands by creating an Oracle Wallet, configuring SSL parameters and restarting Oracle Internet Directory.

This section includes the following topics:

- Configuring SSL Parameters by Using LDAP Commands
- SSL Attributes

28.3.1 Configuring SSL Parameters by Using LDAP Commands

You can configure SSL using LDAP commands.

You must perform the following steps to configure SSL:

1. Create an Oracle wallet.
2. Configure SSL parameters.
3. Restart Oracle Internet Directory.

> **See Also:**
>
> - Managing Oracle Internet Directory Instances for instructions on how to stop and start the server
> - Configuring Certificate Authentication Method by Using Command-Line Tools
> - Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management* for descriptions of the SSL parameters

If you already have created a wallet, you can use the `ldapmodify` command to change SSL parameters. However, you can also create a wallet by using `orapki`. See orapki in *Administering Oracle Fusion Middleware* (which includes Command Line steps to create a Signed Certificate for Testing Purposes).

For example, to change the value of `orclsslinteropmode` to `1` for the instance `oid1`, you would type:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

where `ldifFile` contains:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclsslinteropmode
orclsslinteropmode: 1
```

SSL parameters are attributes of an instance-specific configuration entry. These configuration entries have DNs of the form:

```
cn=componentname,cn=osdldapd,cn=subconfigsubentry
```

for example:

```
cn=oid1,cn=osdldapd,cn=subconfigsubentry
```

The SSL attributes are shown in Table 28-6.

You can use the `ldapsearch` command to list the SSL attributes and their values. For example, to list attributes containing the string `orclssl` in the instance `oid1`, you would type:

```
ldapsearch -p 3060 -D cn=orcladmin -q \
     -b "cn=oid1,cn=osdldapd,cn=subconfigsubentry" \
     -s base "objectclass=*" | grep -i orclssl
```

After you have configured SSL Parameters, restart Oracle Internet Directory, as described in Managing Oracle Internet Directory Instances.

> **Note:**
>
> Set `orclsslenable` to `1` (SSL only) or `2` (Non-SSL & SSL mode) if you use Oracle Enterprise Manager Fusion Middleware Control or WLST to manage Oracle Internet Directory.

28.3.2 SSL Attributes

The SSL attributes are listed in the table with its corresponding meaning.

Table 28-6 lists the SSL Attributes and its meaning:

Table 28-6 SSL Attributes

Attribute	Meaning
`orclsslversion`	SSL Version
`orclsslwalleturl`	SSL Wallet URL
`orclsslenable`	SSL Enable
`orclsslauthentication`	SSL Authentication
`orclsslinteropmode`	SSL Interoperability Mode
`orclsslciphersuite`	SSL Cipher Suite

28.4 Configuring SSL in Oracle Internet Directory

You need to create a SSL connection for Oracle Internet Directory.

To create a SSL connection, complete the following steps:

1. Run the following command to create a wallet with *auto_login* enabled:

   ```
   orapki wallet create -wallet <WALLET_PATH> -auto_login
   ```

2. Run the following command to add a self-signed root certificate inside the created wallet:

   ```
   orapki wallet add -wallet <WALLET_PATH> -dn cn=<OID_HOSTNAME> -keysize 2048 -
   self_signed -validity 3650 -pwd <WALLET_PASSWORD> -sign_alg sha256
   ```

> **Note:**
>
> To add certificates generated by trusted third party CA, see orapki Utility Commands Summary.

3. Modify the OID configuration `cn=oid1,cn=osdldapd,cn=subconfigsubentry` to enable SSL, add cipher suites, and so on.
4. Restart Oracle Internet Directory.
5. Run the following command to test the SSL configuration:

   ```
   ldapbind -h <OID_HOSTNAME> -p <OID_SLL_PORT> -D cn=orcladmin -w
   <OID_ADMIN_PASSWORD> -U 2 -W 'file:<WALLET_PATH>' -P <WALLET_PASSWORD>
   ```

6. In ODSM, create an OID connection by providing the hostname as provided in Step 2.

28.5 Configuring ODSM Connection with SSL Enabled

You can configure ODSM connection to use one-way SSL or two-way SSL for Oracle Internet Directory.

> **Note:**
>
> If Java Development Kit (JDK) 1.8.0_201 or higher is installed on your system then the anonymous ciphers are disabled by default.
>
> If Oracle Internet Directory SSL is configured in SSL No Authentication Mode then you must enable the anonymous ciphers in the JDK by editing the `java.security` file (`JAVA_HOME/lib/security`) and removing `anon` and `NULL` from the `jdk.tls.disabledAlgorithms` security property.

Configure the ODSM wallet by completing the following steps:

> **Note:**
>
> Ensure that the SSL connection for Oracle Internet Directory is created. See Configuring SSL in Oracle Internet Directory.

1. Export the trusted certificate from the Oracle Internet Directory wallet using the `orapki` utility:

   ```
   orapki wallet export -wallet <OID_WALLET_PATH> -dn cn=<OID_HOSTNAME> -cert
   <CERTIFICATE_FILE_NAME>
   ```

   See orapki wallet export in *Administering Oracle Fusion Middleware*.
2. Run the following command to create an ODSM wallet with auto-login enabled:

   ```
   orapki wallet create -wallet <WALLET_PATH> -auto_login
   ```

See orapki wallet create in *Administering Oracle Fusion Middleware*.

> **Note:**

3. Add the trusted certificate that you have exported earlier into the OSDM wallet:

```
orapki wallet add -wallet <WALLET_PATH> -trusted_cert -cert
<CERTIFICATE_FILE_NAME>
```

See orapki wallet add in *Administering Oracle Fusion Middleware*.

28.6 Testing SSL Connections by Using Oracle Directory Services Manager

You can test the SSL connection by using Oracle Directory Services Manager:

Perform the following procedure:

1. Invoke ODSM as described in Invoking Oracle Directory Services Manager.
2. Connect to the Oracle Internet Directory server. On the login screen, enable SSL and specify the SSL port.

If you can connect, the SSL connection is working correctly.

28.7 Overview of Testing SSL Connections From the Command Line

You can use the `ldapbind` command to test SSL connections.

On UNIX, the syntax is:

```
ldapbind -D cn=orcladmin -q -U authentication_mode -h host -p SSL_port \
-W "file://DIRECTORY_CONTAINING_WALLET" -Q
```

and on Windows, the syntax is:

```
ldapbind -D cn=orcladmin -q -U authentication_mode -h host -p SSL_port \
-W "file:device:\DIRECTORY_CONTAINING_WALLET" -Q
```

where *authentication_mode* is one of:

Number	Authentication
1	SSL No authentication required.
2	One-way (server only) SSL authentication required.
3	Two-way (client and server) SSL authentication required.

> **See Also:**
>
> The `ldapbind` command-line tool reference in *Reference for Oracle Identity Management*.

This section includes the following topics:

- Testing SSL With Encryption Only
- Testing SSL With Server Authentication
- Testing SSL With Client and Server Authentication

28.7.1 Testing SSL With Encryption Only

Use this method to test an SSL configuration with SSL no authentication required.

The syntax is:

```
ldapbind -D cn=orcladmin -q -U 1 -h host -p SSL_Port
```

28.7.2 Testing SSL With Server Authentication

Use this method to test an SSL configuration with SSL server authentication configured. A client can request either server authentication or no authentication.

For an anonymous bind with server authentication, the syntax is:

```
ldapbind -U 2 -h host -p SSL_Port -W "file:DIRECTORY_CONTAINING_WALLET" -Q
```

For a bind with user `cn=orcladmin`, wallet file *`$DOMAIN_HOME`*`/config/fmwconfig/components/OID/admin/mywallet`, and server authentication, the syntax is:

```
ldapbind -D cn=orcladmin -q -U 2 -h SSL_Port -p port \
-W "file:$DOMAIN_HOME/config/fmwconfig/components/OID/admin/mywallet" -Q
```

For a bind without SSL authentication, the syntax is:

```
ldapbind -D cn=orcladmin -q -U 1 -h host -p SSL_Port
```

28.7.3 Testing SSL With Client and Server Authentication

Use this method to test an SSL configuration with SSL client and server authentication configured.

Oracle Internet Directory supports the Certificate Matching Rule. The DN and password passed on the `ldapbind` command line are ignored. Only the DN from the certificate or the certificate hash is used for authorization.

> **See Also:**
>
> About Direct Authentication.

To use the bind DN (Distinguished Name) from the client certificate, the syntax is:

```
ldapbind -U 3 -h host -p SSL_Port -W "file:DIRECTORY_CONTAINING_WALLET" -Q
```

28.8 Configuring SSL between Database and Oracle Internet Directory

Use the instructions below to enable SSL connection between Oracle Database and Oracle Internet Directory.

Perform the steps in the following order:

1. Stopping an Instance of Oracle Internet Directory
2. Stopping Node Manager
3. Stopping Administration Server
4. Modifying the sqlnet.ora and listener.ora Files on the Database Server
5. Modifying the tnsnames.ora and sqlnet.ora configuration files on the OID Server
6. Setting the JAVA_OPTIONS Environment Variable on the Administration Server
7. Setting the JAVA_OPTIONS Environment Variable on the Node Manager
8. Restarting an Instance of Oracle Internet Directory

28.8.1 Stopping an Instance of Oracle Internet Directory

You can stop an Oracle Internet Directory (OID) instance using a script.

To stop an OID instance, use the following script:

```
$DOMAIN_HOME/bin/stopComponent.sh <instance_name>
```

28.8.2 Stopping Node Manager

You can stop Node Manager using a script.

To stop Node Manager, use the following script:

```
$DOMAIN_HOME/bin/stopNodeManager.sh
```

28.8.3 Stopping Administration Server

You can stop the Oracle WebLogic Server Administration Server using a script.

To stop an Administration Server, use the following script:

```
$DOMAIN_HOME/bin/stopWebLogic.sh
```

28.8.4 Modifying the sqlnet.ora and listener.ora Files on the Database Server

You must edit the `listener.ora` and `sqlnet.ora` configuration files on the database server to enable SSL communication.

To enable SSL on the database server, perform the following steps:

1. In a terminal, navigate to the following directory:

```
$ cd $DB_HOME/network/admin
```

2. Modify the `listener.ora` file and make sure to add TCPS entry and assign a specific port under the LISTENER section to enable SSL.

```
LISTENER=
 (DESCRIPTION=
    (ADDRESS_LIST=
      (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=ipc)(KEY=extproc))
      (ADDRESS=(PROTOCOL=tcps)(HOST=sales-server)(PORT=1522))))
```

3. Set the database wallet location in the `sqlnet.ora` file of the database Oracle home.

```
wallet_location = (SOURCE= (METHOD=File) (METHOD_DATA= (DIRECTORY=wallet_location)))
```

4. Restart the listener process on the database server for the changes to take effect.

```
lsnrctl stop
lsnrctl start
```

28.8.5 Modifying the tnsnames.ora and sqlnet.ora configuration files on the OID Server

You must edit the `tnsnames.ora` and `sqlnet.ora` configuration files on the OID server to enable SSL communication.

To enable SSL on the OID server, perform the following steps:

1. In a terminal, navigate to the following directory:

```
$ cd $DOMAIN_HOME/config/fmwconfig/components/OID/config
```

2. Edit the `tnsnames.ora` file to specify the database's DN and the TCP/IP with SSL protocol.

```
finance= (DESCRIPTION=(ADDRESS_LIST=(ADDRESS= (PROTOCOL = tcps) (HOST =
finance_server) (PORT = 1575)))(CONNECT_DATA=(SERVICE_NAME= Finance.us.acme.com))
(SECURITY=(SSL_SERVER_CERT_DN="cn=finance,cn=OracleContext,c=us,o=acme"))
```

3. Edit the `sqlnet.ora` file to specify the wallet location.

```
SQLNET.AUTHENTICATION_SERVICES = (BEQ, TCPS)
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_VERSION = 1.2 or 1.1 or 1.0 or 3.0
WALLET_LOCATION =
(SOURCE=
(METHOD=File)
(METHOD_DATA=
(DIRECTORY=wallet_location)))

SSL_SERVER_DN_MATCH=OFF
```

28.8.6 Setting the JAVA_OPTIONS Environment Variable on the Administration Server

You must set the `JAVA_OPTIONS` environment variable to include the wallet information before starting Administration Server from the same terminal.

To set the `JAVA_OPTIONS` environment variable on the Administration Server, perform the following steps:

1. In a terminal, navigate to the following directory:

```
$ cd $DOMAIN_HOME/bin
```

2. Set the `JAVA_OPTIONS` environment variable to include the wallet information.
   - For SSL No Authentication and SSL Server Authentication modes:

     ```
     export JAVA_OPTIONS="-Djavax.net.ssl.trustStore=<wallet_location>/cwallet.sso -
     Djavax.net.ssl.trustStoreType=SSO"
     ```

   - For SSL Mutual Authentication mode:

     ```
     export JAVA_OPTIONS="-Djavax.net.ssl.trustStore=<wallet_location>/cwallet.sso -
     Djavax.net.ssl.trustStoreType=SSO -Djavax.net.ssl.keyStore=<wallet_location>/
     cwallet.sso -Djavax.net.ssl.keyStoreType=SSO"
     ```

> **Note:**
>
> Truststore verifies the server-side certificates, while keystore contains client certificates that are sent to the server.

3. Start the Administration Server.

   ```
   $ ./startWeblogic.sh
   ```

28.8.7 Setting the JAVA_OPTIONS Environment Variable on the Node Manager

You must set the `JAVA_OPTIONS` environment variable to include the wallet information before starting Node Manager from the same terminal.

To set the `JAVA_OPTIONS` environment variable on the Node Manager, perform the following steps:

1. In a terminal, navigate to the following directory:

   ```
   $ cd $DOMAIN_HOME/bin
   ```

2. Set the `JAVA_OPTIONS` environment variable to include the wallet information.
   - For SSL No Authentication and SSL Server Authentication modes:

     ```
     export JAVA_OPTIONS="-Djavax.net.ssl.trustStore=<wallet_location>/cwallet.sso -
     Djavax.net.ssl.trustStoreType=SSO"
     ```

   - For SSL Mutual Authentication mode:

     ```
     export JAVA_OPTIONS="-Djavax.net.ssl.trustStore=<wallet_location>/cwallet.sso -
     Djavax.net.ssl.trustStoreType=SSO -Djavax.net.ssl.keyStore=<wallet_location>/
     cwallet.sso -Djavax.net.ssl.keyStoreType=SSO"
     ```

> **Note:**
>
> Truststore verifies server side certificates, while keystore contains client certificates that are sent to the server.

3. Start the Node Manager.

   ```
   $ ./startNodeManager.sh
   ```

28.8.8 Restarting an Instance of Oracle Internet Directory

You can start an instance of Oracle Internet Directory using a script.

To restart Oracle Internet Directory, use the following script:

```
$DOMAIN_HOME/bin/startComponent.sh <instance-name>
```

29

Configuring Data Privacy

This chapter describes data privacy and explains how Oracle Internet Directory protects data after it is received (since data is protected by SSL during transmission). Specifically, it covers enabling and disabling table space encryption using Oracle Database Transparent Data Encryption (TDE), configuring Oracle Database Vault to protect Oracle Internet Directory data, and configuring sensitive and hashed attributes.
This chapter includes the following sections:

- Introduction to Table Space Encryption
- Enabling and Disabling Table Space Encryption
- Introduction to Using Database Vault With Oracle Internet Directory
- Configuring Oracle Database Vault to Protect Oracle Internet Directory Data
- Best Practices for Using Database Vault with Oracle Internet Directory
- Introduction to Sensitive Attributes
- Enabling Privacy Mode of Sensitive Attributes
- Knowing Privacy Mode Status of Sensitive Attributes
- Introduction to Hashed Attributes
- Configuring Hashed Attributes

29.1 Introduction to Table Space Encryption

Oracle Database Transparent Data Encryption (TDE), a component of Oracle Enterprise User Security, transparently encrypts data when it is written to disk and decrypts it when it is read back to the authorized user. TDE helps protect data stored on media if the storage media or data file gets stolen Applications don't have to be modified, and the data encryption on the storage media is transparent to users.

Oracle Database Advanced Security Transparent Data Encryption introduced support for encryption of database table spaces. All objects created in an encrypted tablespace are automatically encrypted. All data in an encrypted tablespace is stored in encrypted format on the disk. Data blocks are transparently decrypted as they are accessed by the Oracle Database. Table space encryption eliminates the foreign key restriction of column encryption and enables index range scans on encrypted data.

29.2 Enabling and Disabling Table Space Encryption

You can enable or disable table space encryption on Oracle Database used by Oracle Internet Directory.

Refer to the following topics:

- Configuring First-time Settings for Table Space Encryption
- Enabling or Disabling Table Space Encryption

29.2.1 Configuring First-time Settings for Table Space Encryption

If you are trying to configure settings for table space encryption for the first time, follow the procedure as given here.,

If you are enabling table space encryption for the first time:

1. Make a cold backup of the Oracle Databases that are used by the Oracle Internet Directory instances.
2. Make sure you have the JavaVM and XML developer's Kit packages installed in the database Oracle home.

   To verify whether the specified packages are installed, execute the following SQL*Plus:

   ```
   SELECT comp_id, status FROM dba_registry;
   ```

   Execute the following PL/SQL procedure:

   ```
   sys.dbms_metadata_util.load_stylesheets
   ```

3. Log in to SQL*Plus as a user who has the `SYSTEM` privilege and execute the following command:

   ```
   GRANT CREATE ANY DIRECTORY TO ods;
   ```

4. Create the directory object, log directory object used for dumpfiles, and logfiles of the Oracle DataPump utility. Log in to SQL*Plus as the `ODS` user and execute the following commands:

   ```
   CREATE OR REPLACE DIRECTORY directory_object_name as directory_path;
   CREATE OR REPLACE DIRECTORY log_directory_object_name as log_directory_path;
   ```

5. Create *directory_path* and *log_directory_path* in the file system.
6. Set the database wallet location in the `sqlnet.ora` of the database Oracle home.

   > **Note:**
   >
   > Do not confuse the database wallet with the Oracle Internet Directory wallet described in Configuring Secure Sockets Layer (SSL).

   Oracle recommends that you use a separate wallet exclusively for table space encryption.

   a. To use a separate database wallet for table space encryption, set the parameter ENCRYPTION_WALLET_LOCATION in `sqlnet.ora`. For example:

      ```
      ENCRYPTION_WALLET_LOCATION=(SOURCE=(METHOD=FILE)(METHOD_DATA=(DIRECTORY=/install/
      db11g/dbs)))
      ```

   b. To use the same database wallet shared by all Oracle components, set the parameter `WALLET_LOCATION` in `sqlnet.ora`. For example:

      ```
      WALLET_LOCATION= (SOURCE=(METHOD=FILE)(METHOD_DATA=(DIRECTORY=/install/db11g/
      dbs)))
      ```

29.2.2 Enabling or Disabling Table Space Encryption

You can enable or disable table space encryption by following the procedure as given here.,

To enable or disable table space encryption:

1. Shut down all the Oracle Internet Directory instances that are using the Oracle Database Oracle home.
2. If you are enabling table space encryption for the first time in the Oracle Database Oracle home, log in to SQL*Plus as a user who has the `ALTER SYSTEM` privilege and execute the following command:

   ```
   ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY yourwalletpassword;
   ```

3. Whenever the Oracle Database is shut down and restarted, log in to SQL*Plus as a user who has the `ALTER SYSTEM` privilege and execute the following command:

   ```
   ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY yourwalletpassword;
   ```

   Be sure to execute the command before starting Oracle Internet Directory and before running the Perl script shown in Step 6.
4. Set the environment variable `ORACLE_HOME` to the Oracle Database home.
5. Set the environment variable `NLS_LANG` to the character set of the Oracle Database server.
6. Edit the path of the `perl5` executable in the Perl script *`ORACLE_HOME`*`/ldap/datasecurity/oidtbltde.pl` so that it matches the location of `perl5` on your computer.
7. If you have not already done so, install the database independent interface module for Perl (DBI) and the Oracle DBD driver for Perl.
8. Run the Perl script `oidtbltde.pl` to enable or disable TDE for Oracle Internet Directory.

29.3 Introduction to Using Database Vault With Oracle Internet Directory

Oracle Internet Directory enforces access control in the LDAP protocol layer. However, a privileged user such as `DBA` can normally access the Oracle Internet Directory data in the underlying database by using SQL*Plus.

You can use Oracle Database Vault to prevent unauthorized access to Oracle Internet Directory data by a privileged user. To do so, you must install and enable Oracle Database Vault, set up a Database Vault realm containing the `ODS` database schema used by Oracle Internet Directory, and set up a policy to allow only the `ODS` database account to access the data.

See Also:

- *Oracle Database 2 Day + Security Guide* for a quick guide to installing, enabling, and disabling Oracle Database Vault
- *Oracle Database Vault Administrator's Guide* for detailed information about administering Oracle Database Vault

29.4 Configuring Oracle Database Vault to Protect Oracle Internet Directory Data

You must install and register Oracle Database Vault before you configure it for Oracle Internet Directory. You install Database Vault as part of the Oracle Database installation.

The following sections explain this further:

- Registering Oracle Database Vault with Oracle Internet Directory for First Time
- Knowing Whether Oracle Database Vault is Registered with Oracle Database
- Installing Bug Patches for Existing Oracle Database Vault Registration
- Adding Database Vault Realm to Apply Policies
- Enabling SQL*Plus Access to the Oracle Internet Directory Database
- Blocking SQL*Plus Access to the Oracle Internet Directory Database
- Database Vault Rules Defined for Oracle Internet Directory
- Deleting Database Vault Policies For Oracle Internet Directory
- Disabling Oracle Database Vault for the Oracle Internet Directory Database

29.4.1 Registering Oracle Database Vault with Oracle Internet Directory for First Time

If Oracle Database Vault is not registered with your Oracle Database 19c, proceed as follows:

1. Install Oracle Internet Directory as described in Verifying OID Installationin *Installing and Configuring Oracle Internet Directory*.
2. Register Oracle Database Vault as described in Registering Oracle Database Vault with an Oracle Database in *Oracle Database Vault Administrator's Guide*.

29.4.2 Knowing Whether Oracle Database Vault is Registered with Oracle Database

Using SQL command, you can find out if the Oracle database vault is registered with Oracle database.

If you do not know whether Oracle Database Vault was registered with your Oracle Database 19c, type:

```
SELECT * FROM V$OPTION WHERE PARAMETER = 'Oracle Database Vault';
```

at a SQL*Plus prompt. If the query returns `Oracle Database Vault`, then Oracle Database Vault has been installed and registered. Note that the query is case-sensitive.

29.4.3 Installing Bug Patches for Existing Oracle Database Vault Registration

To know how to install bug patches for existing Oracle database vault registration, follow the instructions given here.

If Oracle Database Vault was registered with your Oracle Database, proceed as follows:

1. Disable Oracle Database Vault, if it is enabled. See the appendix titled "Disabling and Enabling Oracle Database Vault" in *Oracle Database Vault Administrator's Guide*.
2. Install Oracle Internet Directory as described in Verifying Oracle Internet Directory Installation in *Installing and Configuring Oracle Internet Directory*.
3. Enable Database Vault as described in theEnabling Oracle Database Vault in *Oracle Database Vault Administrator's Guide*.
4. Download and install the patch for Bug 7244497 if you are using Oracle Database version 19c.
5. Download and install the patch for Bug 7291157 if you are using Oracle Database version 19c.

29.4.4 Adding Database Vault Realm to Apply Policies

Oracle Internet Directory provides scripts to apply the required Database Vault policies. These scripts are located in the Oracle Internet Directory installation under `$ORACLE_HOME/ldap/datasecurity`.

To apply the Database Vault policies to the Oracle Internet Directory database, you must create the default Database Vault realm for Oracle Internet Directory, as follows:

1. Open `dbv_oid_rule.sql` in a text editor and replace the dummy IP address in the `Check ods connections` and `Check ods connections 2` rules with the

   IP address of the computer where Oracle Internet Directory is running.
2. Connect to the database as the Database Vault owner and execute `dbv_create_oid_policies.sql`.

29.4.5 Enabling SQL*Plus Access to the Oracle Internet Directory Database

The policies in `dbv_create_oid_policies.sql` completely disable SQL*Plus access to the Oracle Internet Directory database.

For some tasks, you might require SQL*Plus access to the database by the `ODS` user. If so, enable SQL*Plus access to the Oracle Internet Directory Database from a specific host or hosts only.

To enable connectivity to the Oracle Internet Directory Database, follow these steps:

1. Open `dbv_oid_rule_sqlplus.sql` in a text editor. Replace the dummy IP address in `Check ods connections 3` rule with the IP addresses of the hosts from which to allow SQL*Plus access to Oracle Internet Directory Database.
2. Connect to the database as the Database Vault owner and execute `dbv_oid_rule_sqlplus.sql`.

29.4.6 Blocking SQL*Plus Access to the Oracle Internet Directory Database

You can block SQL*Plus access to Oracle internet directory database by following the command given here.

If you want to block SQL*Plus access completely to the Oracle Internet Directory database at some point, connect to the Database as the Database Vault owner and execute `dbv_oid_delete_rule_sqlplus.sql`.

29.4.7 Database Vault Rules Defined for Oracle Internet Directory

The Database Vault rules defined for Oracle Internet Directory are `Check ods connections`, `Check ods connections 2`, `Check odssm connections`, and `Allow other connections`.

The Configuring Command Rules in *Oracle Database Vault Administrator's Guide* explains how to use data dictionary views. This section describes some views that report Oracle Internet Directory-related information.

The name of the Database Vault realm that Oracle Internet Directory uses is `OID Realm`. You can verify that the realm exists by querying the `DBA_DV_REALM` data dictionary view.

The Database Vault rules defined for Oracle Internet Directory are `Check ods connections`, `Check ods connections 2`, `Check odssm connections`, and `Allow other connections`. If you ran `dbv_oid_rule_sqlplus.sql`, the rule `Check ods connection 3` is also defined. These rules are added to a rule set named `OID App Access`. To check the names of the existing rules, query the `DBA_DV_RULE_SET_RULE` view.

A `CONNECT` command rule is firing this rule set. You can verify this by querying the `DBA_DV_COMMAND_RULE` view. This `CONNECT` rule does not overwrite existing `CONNECT` command rules when you run the Oracle Internet Directory scripts on an existing Oracle Database Vault installation.

29.4.8 Deleting Database Vault Policies For Oracle Internet Directory

You can delete database vault policies for Oracle Internet Directory by using the following command:

To remove the Database Vault policies for OID installed in the prior section, execute `dbv_delete_oid_policies.sql` while connected to the database as the Database Vault Owner.

29.4.9 Disabling Oracle Database Vault for the Oracle Internet Directory Database

You can disable Oracle database vault for Oracle Internet Directory database.

See Disabling and Enabling Oracle Database Vault in *Oracle Database Vault Administrator's Guide*.

29.5 Best Practices for Using Database Vault with Oracle Internet Directory

The following administrative tasks require special attention when Oracle Database Vault is in use:

- Upgrading Products and Installing Patchsets–disable Oracle Database Vault before performing Oracle Internet Directory or Oracle Database upgrades or patchset installations. Enable Oracle Database Vault again after the upgrade or installation is complete.
- Bulk Loading Data–when Oracle Database Vault is enabled, the SQL*Loader direct path mode is unavailable, which reduces the performance of the `bulkload` tool. Disable Oracle Database Vault before using `bulkload` to load more than 100KB of data or more than one million entries. Enable Oracle Database Vault again after the operation is complete.

29.6 Introduction to Sensitive Attributes

Oracle Internet Directory stores sensitive attributes in an encrypted format.

Examples of sensitive attributes are: `orclpasswordattribute`, `orclrevpwd`, the plug-in attribute `orclpluginsecuredflexfield` and the server chaining attribute `orclOIDSCExtPassword`.

The following sections explain sensitive attributes further:

- List of Sensitive Attributes
- Encryption Algorithm for Sensitive Attributes

29.6.1 List of Sensitive Attributes

The list of sensitive attributes is stored in the attribute `orclencryptedattributes` in the DSA configuration entry.

The list is shown in Table 29-1.

Table 29-1 Sensitive Attributes Stored in orclencryptedattributes

Sensitive Attribute	Attribute Usage
`orclpluginsecuredflexfield`	Sensitive attributes passed to a plug-in. See Table 46-1.
`orcloidscextpassword`	Server admin password for plug-in connection. See Table 46-1.
`orcloidscwalletpassword`	Plug-in sslwallet password. See Configuring Directory Server Chaining
`orclrevpwd`	User password in reversible encrypted format. See Managing Password Verifiers .
`orclunsyncrevpwd`	Encrypted reversible password NOT synchronized with the related userpassword. See Managing Password Verifiers .
`orclodipprofileinterfaceconnectinformation`	Oracle Directory Integration Platform: Information used to connect to an application for event propagation.

Table 29-1 (Cont.) Sensitive Attributes Stored in orclencryptedattributes

Sensitive Attribute	Attribute Usage
orclodipcondiraccesspassword	Oracle Directory Integration Platform: Used by third-party directory to connect to directory.
orclodipagentpassword	Oracle Directory Integration Platform: Password that the synchronization profile uses to bind to the directory.

For information about the last three entries, see LDAP Attribute Reference chapter in *Reference for Oracle Identity Management*.

The orcldataprivacymode attribute controls whether these attributes are encrypted when the data is received. When orcldataprivacymode is enabled, the sensitive attributes are encrypted.When privacy mode is disabled, the sensitive data is returned in the clear.

If you add an encrypted attribute to the list of sensitive attributes, you must restart the Oracle Internet Directory server instance for the new attribute to be added to the new list of sensitive attributes and recognized by the server.

> **Note:**
>
> The attributes in Table 29-1 are intended for use only by Oracle. Do not add to or modify the attributes shown in this table unless you are requested to do so by Oracle Support.

29.6.2 Encryption Algorithm for Sensitive Attributes

Prior to 11*g* Release 1 (11.1.1.4.0), Oracle Internet Directory used the 3DES encryption algorithm for the storage of sensitive attributes. The Oracle Internet Directory now uses AES-256.

Customers who have patched their systems from an earlier release might already have stored values encrypted with the 3DES algorithm. In such cases, the following rules apply:

- At decryption time, Oracle Internet Directory uses the appropriate algorithm (3DES or AES-256) to decrypt the value.
- At encryption time, Oracle Internet Directory always encrypts using AES-256.

This ensures that, over time, all encrypted values are converted to AES-256.

29.7 Enabling Privacy Mode of Sensitive Attributes

Privacy mode is disabled by default. That is, the value of orcldataprivacymode is 0.

To provide security protection, you must enable privacy mode by changing the value of orcldataprivacymode from 0 to 1 in the DSA configuration entry

To enable privacy mode, use an LDIF file containing the following entries:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
```

```
replace: orcldataprivacymode
orcldataprivacymode: 1
```

Load the LDIF file with a command line similar to this:

```
$ORACLE_HOME/bin/ldapmodify -h host -p port -D cn=orcladmin -q -v \
   -f LDIF_file_name
```

29.8 Knowing Privacy Mode Status of Sensitive Attributes

To determine the value of `orcldataprivacymode`, perform the following search:

```
$ORACLE_HOME/bin/ldapsearch -h host -p port -D cn=orcladmin -q \
   -b "cn=dsaconfig,cn=configsets,cn=oracle internet directory" -s base \
   "objectclass=*" orcldataprivacymode
```

29.9 Introduction to Hashed Attributes

Unlike encryption, hashing is a one-way operation. It is not possible to derive the original value from the hashed value.

Oracle Internet Directory supports hashed attributes in addition to sensitive attributes. The list of hashed attributes is contained in `orclhashedattributes`, a multivalued attribute of the DSA configuration entry. Hashing is performed using the cryptographic scheme set in the root DSE attribute `orclcryptoscheme`.

LDAP operations and `bulkload` automatically perform the transformations described in Table 29-2. You cannot use the `bulkmodify` command with hashed attributes.

Table 29-2 LDAP and Bulk Operations on Attributes in orclhashedattributes

Operation	When incoming attribute value is already hashed	When incoming attribute value is not yet hashed
`ldapadd`	Use value as it is.	Hash incoming value by using `orclcryptoscheme` before performing operation.
`ldapmodify`	Use value as it is.	For an add or replace operation, hash incoming value by using `orclcryptoscheme` before performing operation. For a delete operation, hash the incoming value using the crypto scheme that was in use at the time the attribute was stored in the directory before performing operation.
`ldapcompare`	Compare incoming value with value stored in directory.	Hash the incoming value by using the crypto scheme that was in use at the time the attribute was stored in the directory and then compare it with the stored value.
`bulkload`	Use value as it is.	Hash incoming value by using `orclcryptoscheme` before performing operation.
`bulkmodify`	Do not allow `bulkmodify`.	Do not allow `bulkmodify`.

> **Note:**
>
> - Never include the same attribute in both `orclhashedattributes` and `orclencryptedattributes`.
> - Only single-valued attributes can be hashed attributes.

29.10 Configuring Hashed Attributes

You can manage the list of attributes in `orclhashedattributes` by using Oracle Enterprise Manager Fusion Middleware Control or the command line.

The following sections explain this configuration further:

- Configuring Hashed Attributes by Using Fusion Middleware Control
- Configuring Hashed Attributes by Using ldapmodify

29.10.1 Configuring Hashed Attributes by Using Fusion Middleware Control

You can configure hashed attributes by using the Shared Properties page in Oracle Enterprise Manager Fusion Middleware Control.

Select **Administration**, then **Shared Properties** from the **Oracle Internet Directory** menu.

29.10.2 Configuring Hashed Attributes by Using ldapmodify

To configure hashed attributes by using the command line, add the attribute names to the DSA configuration entry's multivalued attribute `orclhashedattribute`.

For example, the following LDIF file adds three attributes to `orclhashedattributes`.

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype:modify
add: orclhashedattributes
orclhashedattributes: attributeName1
orclhashedattributes: attributeName2
orclhashedattributes: attributeName3
```

Load the LDIF file with a command line similar to this:

```
$ORACLE_HOME/bin/ldapmodify -h host -p port -D cn=orcladmin -q -v \
   -f LDIF_file_name
```

30
Managing Password Policies

This chapter describes how Oracle Internet Directory manages password policies, which are sets of rules that govern how passwords are used. Specifically, it describes password policies including default policies and fine-grained policies and how to manage password policies using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities. The following sections describe managing password policies:

- Overview of Managing Password Policies
- Managing Password Policies by Using Oracle Directory Services Manager
- Managing Password Policies by Using Command-Line Tools

30.1 Overview of Managing Password Policies

A password policy is a set of rules governing how passwords are used. When a user attempts to bind to the directory, the directory server ensures that the password meets the various requirements set in the password policy.

When you establish a password policy, you set the following types of rules, to mention just a few:

- The maximum length of time a given password is valid
- The minimum number of characters a password must contain
- The number of numeric characters required in a password

This section contains these topics:

- Introduction to Password Policy Rules
- Creating and Applying a Password Policy
- About Fine-Grained Password Policies
- About Default Password Policy
- Attributes for Password Policy
- Operational Attributes of User Entry
- About Directory Server Verification of Password Policy Information
- About Password Policy Error Messages

30.1.1 Introduction to Password Policy Rules

Password polices are sets of rules that govern password syntax and how passwords are used.

Password policies enforced by Oracle Internet Directory include:

- The maximum length of time a given password is valid
- The minimum number of characters a password must contain
- The minimum number of numeric characters required in a password

- The minimum number of alphabetic characters
- The minimum number of repeated characters
- The use of uppercase and lowercase
- The minimum number of non-alphanumeric characters (that is, special characters)
- That users change their passwords periodically
- The minimum and maximum time between password changes
- The grace period for logins after password expiration, by time or by number of logins
- That users cannot reuse previously used passwords

30.1.2 Creating and Applying a Password Policy

In general, establishing a password policy requires the following steps:

1. Create a password policy entry in the appropriate container and associate it with the `pwdpolicy` object. (Default entries exists when you first install Oracle Internet Directory.)
2. Create the desired policy by setting values for attributes defined under the `pwdpolicy` object class for the entry created in step 1.
3. Enable the policy by setting the `orclepwdpolicynable` attribute to 1. If this is not set to 1, Oracle Internet Directory ignores the policy.
4. Determine the subtree to be governed by the policy. Add and populate a `pwdpolicysubentry` attribute with the policy's DN, at the root of that subtree.

> **See Also:**
>
> LDAP Object Class Reference in *Reference for Oracle Identity Management* for a list and descriptions of the attributes of the `pwdPolicy` object class, and those of the `top` object class that pertain to password policies

30.1.3 About Fine-Grained Password Policies

The Oracle Internet Directory supports multiple password policies in each realm. You can apply these policies to any subtree within that realm. This means that you can have entry-specific password policies. You can specify password policies as realm-specific or directory-wide in scope.

To achieve the desired scope, you must create the password policy entry in the appropriate container. Password policies are populated under a "`cn=pwdPolicies`" container created under the "`cn=common`" entry in each realm. By default these containers contain a password policy with the RDN "`cn=default`". The directory specific default password policy, for example, has the DN: `cn=default,cn=pwdPolicies,cn=Common,cn=Products, cn=OracleContext`.

You can create other policies under the `pwdPolicies` container, with different RDNs. Figure 30-1 illustrates this scenario.

Figure 30-1 Location of Password Policy Entries

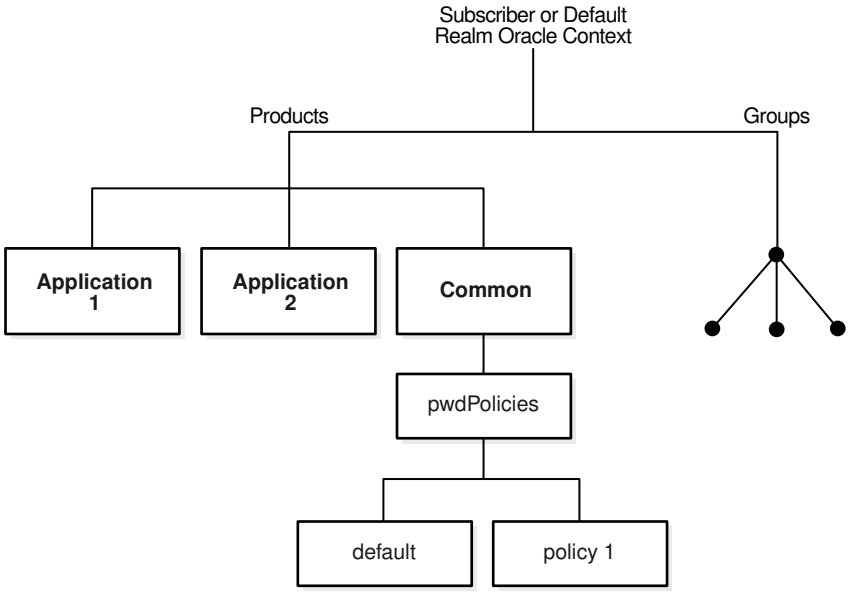
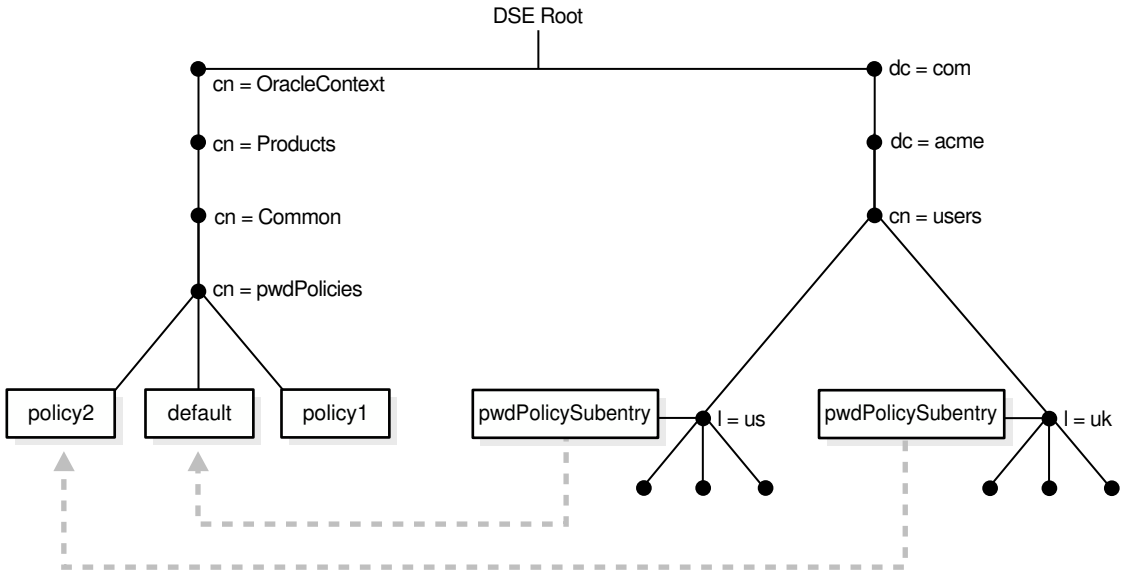


Figure 30-2 pwdPolicy subentry Attributes Populated with DN of Password Policy



At run time, Oracle Internet Directory resolves the applicable password policy on an entry by looking for a populated `pwdpolicysubentry` attribute in the entry and applying the policy pointed to by its value. If a populated `pwdPolicysubentry` attribute does not exist, Oracle Internet Directory traverses up the directory tree until it finds the nearest ancestor entry with a populated `pwdPolicysubentry`. Oracle Internet Directory applies the password policy pointed to by the value at that entry.

> **Note:**
>
> - Password policies applied to groups are **not** automatically applied to group members. You must apply the policy to individual entries or to an ancestor entry.
> - You can disable a password policy by setting `orclpwdpolicyenable` to 0. Doing so leaves that portion of the directory without an applicable password policy. Oracle Internet Directory does not traverse up the DIT to find an enabled policy that is applicable. Setting this attribute to `0` enables you to leave portions of the directory free of password policies when necessary. However you should consider the implications of making such a change before doing so.
> - You must protect password policy entries from anonymous access using Oracle Internet Directory's ACI infrastructure, described in Managing Directory Access Control. This is particularly important when a password policy is weak, as that information can assist an attacker in compromising the directory.

30.1.4 About Default Password Policy

The default password policy for Oracle Internet Directory enforces:

- Password expiration in 120 days
- Account lockout after 10 login failures. Except for the superuser account, all accounts remain locked for a duration of 24 hours unless the passwords are reset by the directory administrator. A user account stays locked even after the lockout duration has passed unless the user binds with the correct password

  If the superuser account, `cn=orcladmin`, becomes locked, it stays locked until you unlock it by using the OID Database Password utility. This utility prompts you for the ODS user password. After you enter the ODS password, it unlocks the account.

  > **See Also:**
  >
  > – The `oidpasswd` command-line tool reference in Oracle Internet Directory Database Password Utility in *Reference for Oracle Identity Management* for information on unlocking a superuser account
  >
  > – Troubleshooting Password Policies.

- A minimum password length of five characters with at least one numeric character
- Password expiry warning seven days before expiry
- Five grace logins allowed after password expiry

Beginning in Oracle Internet Directory, Release 9.0.4, the password policy entry in the Root Oracle Context applies to the superuser, but only the password policy governing account lockout is enforced on that account.

> **Note:**
>
> Oracle Identity Management has two distinct types of privileged user. Both privileged user accounts can be locked if certain password policies are activated.
>
> The first type of privileged user, the superuser with the DN `cn=orcladmin`, is represented as a special user entry found within the default identity management realm. It enables directory administrators to make any modifications to the DIT and any changes to the configuration of Oracle Internet Directory servers. If the superuser (`orcladmin`) account is locked—for example, as a result of too many attempts to bind with an incorrect password—then an administrator with DBA privileges to the Oracle Internet Directory repository can unlock it by using the `oidpasswd` tool. To unlock the orcladmin account execute the command:
>
> ```
> oidpasswd connect="connt_String" unlock_su_acct=TRUE
> ```
>
> The second privileged user, a realm-specific privileged user, governs capabilities such as creation and deletion of users and groups within a realm. This account is represented by an entry with the DN `cn=orcladmin,cn=users,`*`realm DN`*. Note that, in contrast to the single superuser account, each realm has its own realm-specific privileged user. To unlock the realm-specific privileged account, the first type of privileged user, `cn=orcladmin`, can modify the account password by using Oracle Directory Services Manager.

The Oracle Internet Directory password policy is applicable to simple binds (based on the `userpassword` attribute), compare operations on the `userpassword` attribute, and SASL binds. It does not apply to SSL and proxy binds.

30.1.5 Attributes for Password Policy

The attributes that affect the password policy are listed here:

The following attributes affect password policy:

Table 30-1 Password Policy Attributes

Name	Function
`pwdMinAge`	The number of seconds that must elapse between user modifications to the password. The default is 0.
`pwdMaxAge`	The maximum time, in seconds, that a password can be valid. Upon reaching this age, the password is considered to have expired. The default is 10368000 seconds (120 days).
`pwdLockout`	When this is true, the server locks out a user after a number of consecutive invalid login attempts. The number is specified by `pwdMaxFailure`. The default value of `pwdLockout` is 1 (true).
`orclpwdIPLockout`	When this is true, the server locks out a user after a number of consecutive invalid login attempts from the same IP address. The number is specified by `orclpwdIPMaxFailure`.The default is false.
`pwdLockoutDuration`	The time period in seconds to lock out a user account when the threshold of invalid login attempts is reached. The default is 86400 seconds (24 hours).
`orclpwdIPLockoutDuration`	The time period in seconds to lock out a user account when the threshold of invalid login attempts from the same IP address is reached. The default is 0.

Table 30-1 (Cont.) Password Policy Attributes

Name	Function
pwdMaxFailure	The maximum number of invalid login attempts the server should allow before locking out a user account. The default value is 10.
orclpwdIPMaxFailure	The maximum number of invalid login attempts the server should allow from a particular IP address before locking the user account. The default is 0.
pwdFailureCountInterval	The time in seconds after which the password failures are purged from the failure counter, even though no successful authentication occurred. If the value is 0, failure times are never purged. The default is 0.
pwdExpireWarning	The maximum number of seconds before a password is due to expire that expiration warning messages are returned to an authenticating user. The default value is 604800 seconds (seven days).
pwdCheckSyntax	Enables or disables password syntax check 0–Disable all syntax checks 1–Enable password syntax value checks, except for encrypted passwords (default)
pwdMinLength	The minimum length of a password governed by this policy. The default is 5 characters
pwdGraceLoginLimit	The maximum number of grace logins allowed after a password expires. The default is 5. The maximum is 250.
orclpwdGraceLoginTimeLimit	The maximum period in seconds where grace logins are allowed after a password expires. If orclpwdGraceLoginTimeLimit is nonzero, then pwdGraceloginLimit must be zero. If pwdGraceloginLimit is nonzero, then orclpwdGraceLoginTimeLimit must be zero (the default).
pwdMustChange	Requires users to reset their password upon their first login after account creation or after a password has been reset by the administrator. The default is 0 (false).
orclpwdIllegalValues	A list of values that are not allowed as passwords.
orclpwdAlphaNumeric	The minimum number of numeric characters required i in a password. The default is 1.
orclpwdMinAlphaChars	The minimum number of alphabetic characters required in a password. The default is 0.
orclpwdMinSpecialChars	The minimum number of non-alphanumeric characters (that is, special characters) required in a password. The default is 0.
orclpwdMinUppercase	The minimum number of uppercase characters required in a password. The default is 0.
orclpwdMinLowercase	The minimum number of lowercase characters required in a password. The default is 0.
orclpwdMaxRptChars	The maximum number of repeated characters allowed in a password. The default is 0.
pwdInHistory	The maximum number of used passwords stored in the pwdHistory attribute of a given entry. Passwords stored in pwdHistory cannot be used as a new password until they are purged from it. The default is 0.
pwdAllowUserChange	Not currently used.
orclpwdPolicyEnable	When this is true, the server evaluates this policy. Otherwise, the policy is ignored and not enforced. The default is 1 (true).
orclpwdEncryptionEnable	When set to true, enables password encryption. The default is 0 (false).
orclpwdAllowHashCompare	Enables or disables logins using the hashed password value. 0 = disabled (default). 1 = enabled.

Table 30-1 (Cont.) Password Policy Attributes

Name	Function
orclPwdTrackLogin	Enables or disables tracking of user's last login time. 0 = disabled (default). 1= enabled.
orclpwdmaxinactivitytime	Amount of inactive time, in seconds, before an account is automatically expired. 0=disabled (default). The attribute orclPwdTrackLogin must be enabled if orclpwdmaxinactivitytime is non-zero. See Determining Expired Users in Oracle Internet Directory by Using Command-Line Tools.

30.1.6 Operational Attributes of User Entry

The Oracle Internet Directory server stores user-specific password policy-related information in operational attributes of the user entry. Only the server can modify these attributes.

They are shown in Table 30-2.

Table 30-2 Password Policy-Related Operational Attributes

Attribute	Description
orcllastlogintime	Timestamp of last successful login. Tracked only if the password policy attribute orclPwdTrackLogin is enabled.
pwdfailuretime	A space-delimited set of timestamps of failed login attempts, cleared upon successful login.
orclpwdipaccountlockedtime	Time when account was locked for logins from this IP address. This can be a multivalued attribute.
orclpwdipfailuretime	A space-delimited set of timestamps of failed login attempts from a specific IP address, cleared upon successful login. This can be a multivalued attribute.
pwdaccountlockedtime	Time when account was locked.
pwdchangedtime	Time of last password change.
pwdexpirationwarned	Time when user was warned of password expiration.
pwdgraceusetime	A space-delimited set of timestamps of logins during the grace period.
pwdreset	If the value is 1, the user must reset the password at the next login.
pwdhistory	List of previously used passwords.

To determine the last successful login timestamp of a user, tracking of a user's last login time must be enabled. That is, the PwdTrackLogin attribute must be set to 1 for the relevant password policy. This value is not set by default. Then, check the orcllastlogintime attribute of the user entry for the timestamp of the last login.

To determine time of the last login attempt of a user, compare the user's orcllastlogintime attribute with the last timestamp in pwdfailuretime. The most recent of these values is the time of the user's last login attempt.

30.1.7 About Directory Server Verification of Password Policy Information

Oracle Internet Directory determines the applicable policy for an entry by locating the appropriate populated pwdPolicysubentry.

As explained in About Fine-Grained Password Policies, Oracle Internet Directory determines the applicable policy for an entry by locating the appropriate populated `pwdPolicysubentry`. To ensure that the user password meets the requirements of a given policy, the directory server verifies:

- That the password policy is enabled. It does this by checking the value of the attribute `orclpwdpolicyenable` in the password policy entry. A value of 1 indicates that the password policy is enabled. A value of 0 indicates that it is disabled.
- Correctness of password policy syntax information, which includes, for example, the correct number of alphabetic and numeric characters, or the correct password length. The directory server checks the syntax during `ldapadd` and `ldapmodify` operations on the userpassword attribute.
- Password policy state information, which, for example, includes:
  - The timestamp of the user password creation or modification
  - That the minimum password age is greater than the current time minus the time of password creation
  - The timestamp of consecutive failed login attempts by the user
  - The time at which the user account was locked
  - Indicator that the password has been reset and must be changed by the user on first authentication
  - A history of user's previously used passwords
  - Time stamps of grace logins

  If the grace login is set by time period, the server checks the time discrepancy between the current time and the expiration.

  The directory server checks the state information during `ldapbind` and `ldapcompare` operations, but does so only if the `orclpwdpolicyenable` attribute is set to 1.

  To enable password value syntax checking, set the attributes `orclpwdpolicyenable` and `pwdchecksyntax` in the password policy entry to `TRUE`.

30.1.8 About Password Policy Error Messages

Whenever there are password policy violations, the directory server sends to the client various error and warning messages.

In Oracle Internet Directory, the directory server can send these messages as LDAP controls only if the client sends a password policy request control as a part of an LDAP bind or compare operation. If the client does not send the request control, then the directory server does not send the response controls. Instead, it sends errors and warnings as part of additional information.

> **See:**
>
> Troubleshooting Password Policies for a list of the messages and information about how to resolve them

30.2 Managing Password Policies by Using Oracle Directory Services Manager

You can use Oracle Directory Services Manager to create, assign, and modify password policies.

This section describes managing password policies:

- Viewing Password Policies by Using Oracle Directory Services Manager
- Modifying Password Policies by Using Oracle Directory Services Manager
- Creating a Password Policy and Assigning it to a Subtree by Using ODSM

30.2.1 Viewing Password Policies by Using Oracle Directory Services Manager

To view password policies by using Oracle Directory Services Manager, perform the following steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Expand **Password Policy** in the left pane. All of the password policies appear in the left pane, listed by relative DN. Mouse over an entry to see the full DN.
4. Select a password policy to display its information in the right pane.

30.2.2 Modifying Password Policies by Using Oracle Directory Services Manager

To modify the password policies, perform the following steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Expand **Password Policy** in the left pane. All of the password policies appear in the left pane.
4. Select the password policy you want to modify. Five tab pages appear in the right pane.
5. In the **General** tab page, modify the editable attribute fields as needed.
6. Select the **Account Lockout** tab page and, to modify the fields, select **Global Lockout**. Modify the editable attribute fields as needed.
7. Select the **IP Lockout** tab page and, to modify the fields, select **IP Lockout**. Modify the editable attribute fields as needed.
8. Select the **Password Syntax** tab page and, to modify the fields, select **Check Password Syntax**. Modify the editable attribute fields as needed.
9. Select the **Effective Subtree** tab page to modify the subtree to which the policy applies. To add a subtree, select the **Add** icon. Either enter the DN, or select **Browse**, then use the

Select Distinguished Name (DN) Path window to navigate to the subtree to which you want the policy to apply.

10. When you are finished, choose **Apply**.

30.2.3 Creating a Password Policy and Assigning it to a Subtree by Using ODSM

To create a new password policy, perform the following steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Expand **Password Policy** in the left pane. All of the password policies appear in the left pane.
4. To create a new policy, select **Create**. Alternatively, select an existing password policy in the left pant and select **Create Like**.
5. In the **General** tab page, set or modify the editable attribute fields as needed.
6. Select the **Account Lockout** tab page and, to modify the fields, select **Global Lockout**. Modify the editable attribute fields as needed.
7. Select the **IP Lockout** tab page and, to modify the fields, select **IP Lockout**. Modify the editable attribute fields as needed.
8. Select the **Password Syntax** tab page and, to modify the fields, select **Check Password Syntax**. Modify the editable attribute fields as needed.
9. To assign the password policy to a subtree, select the **Effective Subtree** tab page, then select **Add**. Either enter the DN, or select **Browse**, then use the **Select Distinguished Name (DN) Path** window to navigate to the subtree to which you want the policy to apply.
10. When you are finished, choose **Apply**.

30.3 Managing Password Policies by Using Command-Line Tools

This section describes managing password policies using command-line tools in detail:

- Viewing Password Policies by Using Command-Line Tools
- Creating a New Password Policy by Using Command-Line Tools
- Applying a Password Policy to a Subtree by Using Command-Line Tools
- Setting Password Policies by Using Command-Line Tools
- Making a Password Policy Entry Specific by Using Command-Line Tools
- Determining Expired Users in Oracle Internet Directory by Using Command-Line Tools

30.3.1 Viewing Password Policies by Using Command-Line Tools

The following example retrieves password policies under a specific password policy container:

```
ldapsearch -p port -h host \
           -b "cn=pwdPolicies,cn=common,cn=products,cn=OracleContext, \
               o=my_company,dc=com" \
           -s sub "(objectclass=pwdpolicy)"
```

The following example retrieves all password policy entries:

```
ldapsearch -p port -h host -b " " -s sub "(objectclass=pwdpolicy)"
```

30.3.2 Creating a New Password Policy by Using Command-Line Tools

You create a new password policy by adding a policy entry to the appropriate container.

A good way to do this is as follows:

1. Dump the contents of the default entry, `cn=default,cn=pwdPolicies,cn=Common,cn=Products, cn=OracleContext`, to an LDIF file, using `ldapmodify`. For example:

   ```
   ldapsearch -p port -h host -D cn=orcladmin -q -L \
      -b 'cn=default,cn=pwdPolicies,cn=Common,cn=Products, cn=OracleContext' \
      -s base '(objectclass=pwdpolicy)' >> pwdpolicy.ldif
   ```

   As an alternative to `ldapsearch`, you could use `ldifwrite`. Ensure `DOMAIN_HOME` is set, then type:

   ```
   ldifwrite connect="conn_str" \
     baseDN="cn=default,cn=pwdPolicies,cn=Common,cn=Products, cn=OracleContext" \
     ldiffile="pwpolicy.ldif"
   ```

2. Modify the LDIF file so that it has the common name and desired values for the new policy. For example, you might change `cn=default` to `cn=policy1` and change `pwdMaxFailure` from `10` to `5`.

3. Add the new entry by using `ldapadd`. You would use a command line of the form:

   ```
   ldapadd -p port_number -h host -D cn=orcladmin -q -f pwdpolicy.ldif
   ```

30.3.3 Applying a Password Policy to a Subtree by Using Command-Line Tools

You can use Command-Line Tools to apply password policy to a subtree.

To apply the new password policy to the subtree "dn: cn=accounting,c=us" you would use a command line such as:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -f my_file.ldif
```

with an LDIF file such as this:

```
dn: cn=accounting,c=us
changetype: modify
replace: pwdPolicysubentry
pwdPolicysubentry:cn=policy1,cn=pwdPolicies,cn=common,cn=products,
 cn=OracleContext,o=my_company,dc=com
```

30.3.4 Setting Password Policies by Using Command-Line Tools

The following example disables the `pwdLockout` attribute in the default password policy.

The following example disables the `pwdLockout` attribute in the default password policy. It changes the attribute from its default setting of `1` to `0`.

The file `my_file.ldif` contains:

```
dn: cn=default,cn=pwdPolicies,cn=common,cn=products,cn=OracleContext,
 o=my_company,dc=com
changetype:modify
replace: pwdlockout
pwdlockout: 0
```

The following command loads this file into the directory:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -f my_file.ldif
```

The following example modifies pwdMaxAge in the default password policy entry.

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -q -f file
```

where *file* contains:

```
dn: cn=default,cn=pwdPolicies,cn=common,cn=products,cn=OracleContext,
 o=my_company,dc=com
changetype: modify
replace: pwdMaxAge
pwdMaxAge: 10000
```

30.3.5 Making a Password Policy Entry Specific by Using Command-Line Tools

If the password policy is reset for a large number of users, Oracle Internet Directory server must refresh its `passwordPolicySubentry` cache, which can affect performance by causing a large number of SQL query requests to the Oracle database.

You can make a password policy entry specific by subtyping the `entrylevel`. For example, the following command adds a password policy to `A_user`:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -q -f pwdpolicy.ldif
```

where `pwdpolicy.ldif` contains:

```
dn: A_user,cn=users,dc=us,dc=mycompany,dc=com
changetype: modify
add: pwdpolicysubentry;entrylevel
pwdpolicysubentry;entrylevel: cn=pwdpolicies,dc=us,dc=mycompany,dc=com
```

The password policy applies only to `A_user`. If the `entrylevel` subtype is missing in the entry for the `pwdpolicysubentry` attribute, then the password policy applies to the entire subtree.

30.3.6 Determining Expired Users in Oracle Internet Directory by Using Command-Line Tools

In some situations, you might want to determine expired users and then take a specific action, such as deleting those users from the directory.

> **Note:**
>
> Oracle Internet Directory expired users are not indicated by a specific attribute. An expired user is in a transient state that depends on the system time, the maximum inactive time allowed, and the user's last successful login time. The expired state is determined during a bind or password compare operation for the user.

To determine the expired users, your Oracle Internet Directory deployment must be configured as follows:

- The tracking of each user's last successful login time must be enabled by setting the `orclPwdTrackLogin` attribute to 1.
- The `orclpwdmaxinactivitytime` attribute must be set to a value other than 0 (the default). This attribute specifies the inactive time in seconds before a user's account is automatically considered to be expired.

To determine if a user's account is considered to be expired:

1. Determine the time stamp of the user's last successful login from the `orcllastlogintime` attribute. For example:

   ```
   ldapsearch -h oid_host p oid_port -D "cn=orcladmin" -q -s base \
   -b "cn=jdoe,cn=users,o=oracle" "(objectclass=*)" orcllastlogintime
   ```

2. Subtract the user's `orcllastlogintime` value from the current system time. If the result is greater than the `orclpwdmaxinactivitytime` value, then the user is considered to be in the expired state.
3. If you wish, delete the expired user from the directory.

31
Managing Directory Access Control

This chapter provides an overview of Oracle Internet Directory access control policies, including using access control lists (ACLs), access control items (ACIs), and access control policy points (ACPs). Specifically, it describes how to administer directory access control using either Oracle Directory Services Manager (ODSM) or the `ldapmodify` command-line utility. This chapter includes the following sections:

- Overview of Directory Access Control
- Managing Access Control by Using Oracle Directory Services Manager
- Managing Access Control by Using Command-Line Tools

> **See Also:**
>
> – Security Features in Oracle Internet Directory and Managing Authentication for a conceptual explanation before you begin implementing and administering access control policies
>
> – The Access Control Directive Format for information about the format or syntax of Access Control Items (ACIs)

31.1 Overview of Directory Access Control

Authorization is the permission given to a user, program, or process to access an object or set of objects.

When directory operations are attempted within a directory session, the directory server ensures that the user has the permissions to perform those operations. If the user does not have the permissions, then the directory server disallows the operation. The directory server protects directory data from unauthorized operations by directory users by using access control information. Access Control Information and directory access control features are described in the following sections

- About Access Control Information
- Introduction to Access Control Features
- About Access Control Management Constructs
- About Access Control Information Components
- Access Level Requirements for LDAP Operations
- ACL Evaluation

31.1.1 About Access Control Information

Access control information is the directory metadata that captures the administrative policies relating to access control.

The access control information is stored in Oracle Internet Directory as user-modifiable configuration attributes, each of which is called an access control item (ACI).

Typically, a list of these ACI attribute values, called an access control list (ACL), is associated with directory objects. The attribute values on that list represent the permissions that various directory user entities (or subjects) have on a given object.

An ACI consists of:

- The object to which you are granting access
- The entities or subjects to whom you are granting access
- The kind of access you are granting

Access control policies can be prescriptive, that is, their security directives can be set to apply downward to all entries at lower positions in the directory information tree (DIT). The point from which such an access control policy applies is called an access control policy point (ACP).

ACIs are represented and stored as text strings in the directory. These strings must conform to a well-defined format, called the ACI directive format. Each valid value of an ACI attribute represents a distinct access control policy.

31.1.2 Introduction to Access Control Features

The following features of directory access control can be used by applications running in a hosted environment.

- Prescriptive access control

  Enables the service provider to specify access control lists (ACLs) for a collection of directory objects, instead of having to state the policies for each individual object. This feature simplifies the administration of access control, especially in large directories where many objects are governed by identical or similar policies.

- Hierarchical access control administration model

  Enables the service provider to delegate directory administration to hosted companies. The realm could in turn delegate further if necessary.

- Administrative override control for delegated domains

  Enables the service provider to perform diagnosis and recovery from unintentional account lockout or accidental security exposure.

- Dynamic evaluation of access control entities

  Enables subtree administrators to identify both subjects and objects in terms of their namespace and their association with other objects in the directory. For example, the administrator of one realm can allow only a user's manager to update that user's salary attribute. The administrator of another realm can establish and enforce a different policy regarding salary attributes.

You manage access control policies by configuring the values of the ACI attributes within appropriate entries. You can do this by using either Oracle Directory Services Manager or `ldapmodify`.

31.1.3 About Access Control Management Constructs

This section introduces you to structures used for access control in Oracle Internet Directory.

This section discusses the structures used for access control in Oracle Internet Directory:

- About Access Control Policy Points (ACPs)
- About orclACI Attribute for Prescriptive Access Control
- About orclEntryLevelACI Attribute for Entry-Level Access Control
- About Security Groups

31.1.3.1 About Access Control Policy Points (ACPs)

ACPs are entries in which the `orclACI` attribute has been given a value. The `orclACI` attribute value represents the access policies that are inherited by the subtree of entries starting with the ACP as the root of the subtree. When a hierarchy of multiple ACPs exists in a directory subtree, a subordinate entry in that subtree inherits the access policies from all of the superior ACPs. The resulting policy is an aggregation of the policies within the ACP hierarchy above the entry.

For example, if an ACP is established in the HR department entry, and the Benefits, Payroll, and Insurance groups are entries within the HR department, then any entry within those groups inherits the access rights specified in the HR department entry.

When there are conflicting policies within a hierarchy of ACPs, the directory applies well-defined precedence rules in evaluating the aggregate policy.

> **See Also:**
>
> ACL Evaluation.

31.1.3.2 About orclACI Attribute for Prescriptive Access Control

The `orclACI` attribute contains access control list (ACL) directives that are prescriptive—that is, these directives apply to all entries in the subtree below the ACP where this attribute is defined. Any entry in the directory can contain values for this attribute. Access to this attribute itself is controlled in the same way as access to any other attribute.

> **Note:**
>
> It is possible to represent ACL directives specific to a single entry in the `orclACI` attribute. However, in such scenarios, for administrative convenience and performance advantages, Oracle recommends using `orclEntryLevelACI`—discussed in About orclEntryLevelACI Attribute for Entry-Level Access Control. This is because the LDAP configuration overhead increases with the number of directives represented through `orclACI`. You can reduce this overhead by moving entry specific directives from `orclACI` to `orclEntryLevelACI`.

31.1.3.3 About orclEntryLevelACI Attribute for Entry-Level Access Control

When a policy pertains only to a specific entity—for example, a special user—you can maintain the ACL directives within the entry for that entity. You do this by using a user-modifiable configuration attribute called `orclEntryLevelACI`. This attribute contains ACL directives only for the entry with which it is associated.

Any directory entry can optionally carry a value for this attribute. This is because Oracle Internet Directory extends the abstract object class `top` to include `orclEntryLevelACI` as an optional attribute.

The `orclEntryLevelACI` attribute is multi-valued and has a structure similar to that of `orclACI`.

> **See Also:**
>
> Access Control to Objects for the structure definition of the `orclEntryLevelACI` attribute

31.1.3.4 About Security Groups

Group entries in Oracle Internet Directory are associated with either the `groupOfNames` or the `groupOfUniqueNames` object class. Membership in the group is specified as a value of the `member` or `uniqueMember` attribute respectively.

To specify access rights for a group of people or entities, you identify them in security groups. There are two types of security groups: ACP groups and privilege groups.

- About ACP groups
- About Privilege Groups
- About Users in Both Types of Groups
- About Constraints on Security Groups
- Granting Access Rights to a Group
- About Security Group Membership
- Security Group Membership Computing

31.1.3.4.1 About ACP groups

If an individual is a member of an ACP group, then the directory server simply grants to that individual the privileges associated with that ACP group.

Use ACP groups to resolve access at the level of an ACP. For example, suppose you want to give to several hundred users access to browse an entry. You could assign the browse privilege to each entry individually, but this could require considerable administrative overhead. Moreover, if you later decide to change that privilege, you would have to modify each entry individually. A more efficient solution is to assign the privilege collectively. To do this, you create a group entry, designate it as an ACP group, assign the desired privilege to that group, then assign users as members of that group. If you later change the access rights, you must do it in one place, for the group, rather than for each individual user. Similarly, you can remove that privilege from multiple users by removing them from the group, rather than having to access multiple individual entries.

ACP groups are associated with the `orclacpgroup` object class.

31.1.3.4.2 About Privilege Groups

A privilege group is a higher-level access group. It is similar to an ACP group in that it lists users with similar rights. However, it also provides for additional checking beyond a single ACP, as follows: if an ACP denies access, an attribute in the user's entry tells the directory server

whether the user being denied is in any privilege group. If so, then this user has additional rights at a higher administration level, and all higher administration levels in the DIT are checked. If the directory server finds a higher ACP that grants to the privilege group access to the requested object, then it overrides the denials by the subordinate ACP, and grants access to the user. If, however, the `orclACI` or `orclEntryLevelACI` attribute of a subordinate ACP contains the keyword `DenyGroupOverride`, the higher level ACP does not override the subordinate ACP. Use `DenyGroupOverride` to restrict superuser access through privileged groups.

Normally, you would implement only ACP groups. The additional checking that privilege groups provide can degrade performance. Use privilege groups only when access control at higher levels needs the right to override standard controls at lower levels.

Use privilege groups to grant access to administrators who are not recognized by ACPs lower in the DIT. For example, suppose that the global administrator in a hosted environment must perform operations in a realm. Because the global administrator's identity is not recognized in the realm of the hosted company, the directory server, relying only on the ACPs in that realm, denies the necessary access. However, if the global administrator is a member of a privilege group, then the directory server looks higher in the DIT for an ACP that grants to this privilege group the access rights to that subtree. If it finds such an ACP, then the directory server overrides the denials by ACPs in the hosted company's realm.

Add the `DenyGroupOverride` keyword to an ACI to deny access to members of privileged groups.

Privilege groups are associated with the `orclPrivilegeGroup` object class.

31.1.3.4.3 About Users in Both Types of Groups

If a user is a member of both an ACP group and a privilege group, then the directory server performs an evaluation for each type of group. It resolves access rights for the privilege group by looking to ACPs higher in the DIT.

31.1.3.4.4 About Constraints on Security Groups

Do not create a security group that is of both object classes `orclacpgroup` and `orclPrivilegeGroup`.

In nested security groups, the parent group and child group must always be of the same objectclass either `orclacpgroup` or `orclPrivilegeGroup`.

Violating either of these constrains can result in non-deterministic ACI evaluation.

31.1.3.4.5 Granting Access Rights to a Group

To grant access rights to a group of users, you do the following:

1. Create a group entry in the usual way.
2. Associate the group entry with either the `orclPrivilegeGroup` object class or the `orclACPgroup` object class.
3. Specify the access policies for that group.
4. Assign members to the group.

31.1.3.4.6 About Security Group Membership

Entries can have either direct memberships in groups, or indirect memberships in other ACP or privilege groups by means of nested groups, thus forming a forest of privilege groups. Access policies specified at a given level are applicable to all the members directly or indirectly below that level.

Because Oracle Internet Directory evaluates for access control purposes only security groups, it does not allow setting access policies for other types of groups. When a user binds with a specific distinguished name (DN), Oracle Internet Directory computes the user's direct membership in security groups. If it knows the first level groups for the given DN, Oracle Internet Directory computes nesting of all these first level groups into other security groups. This process continues until there are no more nested groups to be evaluated.

Each security group, nested or otherwise, must be associated with a security group object class—either `orclACPgroup` or `orclPrivilegeGroup`. Even if a group is a member of a security group, the directory server does not consider it for access control purposes unless it is associated with a security group object class. When it has determined the user's membership in security groups, the directory server uses that information for the lifetime of the session.

31.1.3.4.7 Security Group Membership Computing

For example, consider the sample group of entries in Table 31-1, each of which, except Group 4, is marked as a privilege group (`objectclass:orclprivilegegroup`). You can set access control policies that apply to the members of group1, group2, and group3.

Table 31-1 Sample Security Groups

Group	Entry

Group 1	`dn: cn=group1,c=us`
	`cn: group1`
	`objectclass: top`
	`objectclass: groupofUniqueNames`
	`objectclass: orclPrivilegeGroup`
	`uniquemember: cn=mary smith,c=us`
	`uniquemember: cn=bill smith,c=us`
	`uniquemember: cn=john smith,c=us`

Group 2	`dn: cn=group2,c=us`
	`cn: group2`
	`objectclass: top`
	`objectclass: groupofUniqueNames`
	`objectclass: orclPrivilegeGroup`
	`uniquemember: cn=mary jones,c=us`
	`uniquemember: cn=joe jones,c=us`
	`uniquemember: cn=bill jones,c=us`
	`uniquemember: cn=john smith,c=us`

Table 31-1 (Cont.) Sample Security Groups

Group	Entry
Group 3	dn:cn=group3,c=us cn: group3 objectclass: top objectclass: groupofUniqueNames objectclass: orclPrivilegeGroup uniquemember: cn=group2,c=us uniquemember: cn=group1,c=us uniquemember: cn=group4,c=us
Group 4	dn: cn=group4,c=us cn: group4 objectclass: top objectclass: groupofUniqueNames uniquemember: cn=john doe,c=uk uniquemember: cn=jane doe,c=uk uniquemember: cn=anne smith,c=us

Group 3 contains the following nested groups:

- `cn=group2,c=us`
- `cn=group1,c=us`
- `cn=group4,c=us`

Access control policies for Group 3 are applicable to members of Group 3, Group 1, and Group 2 because each of them is marked as a privilege group. These same access control policies are not applicable to the members of Group 4 because Group 4 is not marked as a privilege group.

For example, suppose that the user binds to Oracle Internet Directory as a member of Group 4 with the DN `cn=john doe,c=uk`. None of the access policies applicable to the members of Group 3 apply to this user. This is because his only direct membership is to a non-privilege group. By contrast, if the user were to bind as `cn=john smith,c=us`—that is, as a member of Group 1 and Group 2—then his access rights are governed by access policies set up for members of Group 1, Group 2, and Group 3 (in which Group 1 and Group 2 are nested). This is because all three groups are associated with the object class `orclPrivilegeGroup`.

> **See Also:**
>
> Either Managing Group Entries by Using Oracle Directory Services Manager or Managing Group Entries by Using the Command Line for instructions on how to modify a group entry to associate it with or disassociate it from either the `orclPrivilegeGroup` or the `orclACPgroup` object class

31.1.4 About Access Control Information Components

Access control information represents the permissions that various entities or subjects have to perform operations on a given object in the directory.

Thus, an ACI consists of three components:

- Access Control to Objects
- Access Control to an Entity
- Access Control to Operations

31.1.4.1 Access Control to Objects

The *object* part of the access control directive determines the entries and attributes to which the access control applies. It can be either an entry or an attribute.

Entry objects associated with an ACI are implicitly identified by the entry or the subtree where the ACI itself is defined. Any further qualification of objects at the level of attributes is specified explicitly in the ACL expressions.

In the `orclACI` attribute, the entry DN component of the object of the ACI is implicitly that of all entries within the subtree starting with the ACP as its topmost entry. For example, if `dc=com` is an ACP, then the directory area governed by its ACI is:

```
.*, dc=com.
```

However, since the directory area is implicit, the DN component is neither required nor syntactically allowed.

In the `orclEntryLevelACI` attribute, the entry DN component of the object of the ACL is implicitly that of the entry itself. For example, if `dc=example,dc=com` has an entry level ACI associated with it, then the entry governed by its ACI is exactly: `dc=example,dc=com`. Since it is implicit, the DN component is neither required nor syntactically allowed.

The object portion of the ACL allows entries to be optionally qualified by a filter matching some attribute(s) in the entry:

```
filter=(ldapFilter)
```

where *ldapFilter* is a string representation of an LDAP search filter. The special entry selector `*` is used to specify all entries.

Attributes within an entry are included in a policy by including a comma-delimited list of attribute names in the object selector.

```
attr=(attribute_list)
```

Attributes within an entry are excluded from a policy by including a comma-delimited list of attribute names in the object selector.

```
attr!=(attribute_list)
```

The *object* part of an access control directive may also include special keywords. These are:

- `DenyGroupOverride`, which prevents access from being overridden by higher level ACPs
- `AppendToAll`, which causes the subject of an ACI to be added to all other ACIs in that ACP during evaluation.

> **Note:**
>
> Access to the entry itself must be granted or denied by using the special object keyword `ENTRY`. Note that giving access to an attribute is not enough; access to the entry itself through the `ENTRY` keyword is necessary.

> **See Also:**
>
> The Access Control Directive Format for information about the format or syntax of ACIs

31.1.4.2 Access Control to an Entity

This section describes the entity, authentication mode and object constraints.

- About Entity
- Bind Mode
- Bind IP Filter
- Added Object Constraint

31.1.4.2.1 About Entity

Access is granted to entities, not entries. The entity component identifies the entity or entities being granted access.

You can specify entities either directly or indirectly.

Directly specifying an entity: This method involves entering the actual value of the entity, for example group=managers. You can do this by using:

- The wildcard character (*), which matches any entry
- The keyword `SELF`, which matches the entry protected by the access
- The keyword `SuperUser`, which matches the `SuperUser` DN specified in the directory.
- A regular expression, which matches an entry's distinguished name, for example, dn=regex
- The members of a privilege group object: `group=dn`

Indirectly specifying an entity: This is a dynamic way of specifying entities. It involves specifying a DN-valued attribute that is part of the entry to which you are granting access. There are three types of DN-valued attributes:

- `dnattr`: Use this attribute to contain the DN of the entity to which you are granting or denying access for this entry.
- `groupattr`: Use this attribute to contain the DNs of the administrative groups to which you are granting or denying access for this entry.
- `guidattr`: Use this attribute to contain the global user identifier (orclGUID) of the entry to which you want to grant or deny access for this entry.

For example, suppose you want to specify that Anne Smith's manager can modify the salary attribute in her entry. Instead of specifying the manager DN directly, you specify the DN-valued attribute: `dnattr=manager`. Then, when John Doe seeks to modify Anne's salary attribute, the directory server:

1. Looks up the value for her `manager` attribute and finds it to be John Doe
2. Verifies that the bind DN matches the `manager` attribute
3. Assigns to John Doe the appropriate access

31.1.4.2.2 Bind Mode

The bind mode specifies the methods of authentication and of encryption to be used by the subject.

There are four authentication modes:

- MD5Digest
- PKCS12
- Proxy
- Simple: Simple password-based authentication

There are three encryption options:

- SASL
- SSL No Authentication
- SSL One Way

Specifying the encryption mode is optional. If it is not specified, then no encryption is used, unless the selected authentication mode is PKCS12. Data transmitted by using PKCS12 is always encrypted.

There is a precedence rule among authentication choices, and it is as follows:

Anonymous < Proxy < Simple < MD5Digest < PKCS12

This rule means that:

- Proxy authentication blocks anonymous access
- Simple authentication blocks both Proxy and Anonymous access
- MD5Digest authentication blocks Simple, Proxy and Anonymous access
- PKCS12 authentication blocks MD5Digest, Simple, Proxy and Anonymous access

The bind mode syntax is:

```
BINDMODE =(LDAP_AUTHENTICATION_CHOICE + [ LDAP_ENCRYPTION_CHOICE ] )
LDAP_AUTHENTICATION_CHOICE = Proxy | Simple | MD5Digest | PKCS12
LDAP_ENCRYPTION_CHOICE = SSLNoAuth | SSLOneway | SASL
```

The *LDAP_ENCRYPTION_CHOICE* is an optional parameter. If you do not specify it, then the directory server assumes that no encryption is to be used.

31.1.4.2.3 Bind IP Filter

IP address or IP address range of the subject, defined using a standard LDAP filter on the `orclipaddress` attribute. If the subject's IP address matches the filter defined, then the ACI on which it is defined is applicable for that operation.

The bind ip filter syntax is:

```
BINDIPFILTER =(LDAPFILTER_FOR_ORCLIPADDRESS)
```

For example:

The filter `(|(orclipaddress=1.2.3.*)(orclipaddress=1.2.4.*))` applies when the subject's IP address begins with `1.2.3` or begins with`1.2.4`.

The filter `(&(orclipaddress!=1.2.*)(orclipaddress!=3.4.*))` applies when the subject's IP address does not begin with **1.2** and does not begin with **3.4**.

31.1.4.2.4 Added Object Constraint

When a parent entry has *add* access, it can add objects as entries lower in the hierarchy. The added object constraint can be used to limit that right by specifying an *ldapfilter*.

> **See Also:**
>
> The Access Control Directive Format and The LDAP Filter Definition

31.1.4.3 Access Control to Operations

The kind of access granted can be one of the following:

- None
- Compare/nocompare
- Search/nosearch
- Read/noread
- Selfwrite/noselfwrite
- Write/nowrite
- Add/noadd
- Proxy/noproxy
- Browse/nobrowse
- Delete/nodelete

Note that each access level can be independently granted or denied. The `no`*xxx* means *xxx* permission is denied.

Note also that some access permissions are associated with entries and others with attributes.

Table 31-2 Types of Access

Access Level	Description	Type of Object
Compare	Right to perform compare operation on the attribute value	Attributes
Read	Right to read attribute values. Even if read permission is available for an attribute, it cannot be returned unless there is browse permission on the entry itself.	Attributes
Search	Right to use an attribute in a search filter	Attributes

Table 31-2 (Cont.) Types of Access

Access Level	Description	Type of Object
Selfwrite	Right to add yourself to, delete yourself from, or modify your own entry in a list of DNs group entry attribute. Use this to allow members to maintain themselves on lists. For example, the following command allows people within a group to add or remove only their own DN from the member attribute: `access to attr=(member) by dnattr=(member) (selfwrite)` The `dnattr` selector indicates that the access applies to entities listed in the member attribute. The `selfwrite` access selector indicates that such members can add or delete only their own DN from the attribute.	Attributes
Write	Right to modify/add/delete the attributes of an entry.	Attributes
None	No access rights. The effect of granting no access rights to a subject-object pair is to make the directory appear to the subject as though the object were not present in the directory.	Both entries and attributes
Add	Right to add entries under a target directory entry	Entries
Proxy	Allows the subject to impersonate another user	Entries
Browse	Permission to return the DNs in the search result. It is equivalent to the list permission in X.500. This permission is also required for a client to use an entry DN as the base DN in an ldapsearch operation.	Entries
Delete	Right to delete the target entry	Entries

The entry level access directives are distinguished by the keyword `ENTRY` in the object component.

> **Note:**
>
> The default access control policy grants the following to both entries and attributes: Everyone is given access to read, search, write, and compare all attributes in an entry, and selfwrite permissions are unspecified. If an entry is unspecified, access is determined at the next highest level in which access is specified.

31.1.5 Access Level Requirements for LDAP Operations

The table provides information on Access level requirements for LDAP.

Table 31-3 lists the various LDAP operations and the access required to perform each one.

Table 31-3 LDAP Operations and Access Needed to Perform Each One

Operation	Required Access
Create an object	Add access to the parent entry
Modify	Write access to the attributes that are being modified
ModifyDN	Delete access to the current parent and Add access to the new parent
ModifyDN (RDN)	Write access to the naming attribute—that is, the RDN attribute
Remove an object	Delete access to the object being removed

Table 31-3 (Cont.) LDAP Operations and Access Needed to Perform Each One

Operation	Required Access
Compare	Compare access to the attribute and Browse access to the entry
Search	• Search access on the filter attributes and Browse access on the entry (if only the entry DN must be returned as a result) • Search access on the filter attributes, Browse access on the entry, and Read permission on the attributes (for all attributes whose values must be returned as a result)

31.1.6 ACL Evaluation

When a user tries to perform an operation on a given object, the directory server determines whether that user has the appropriate access to perform that operation on that object. If the object is an entry, it evaluates the access systematically for the entry and each of its attributes.

Evaluating access to an object—including an attribute of an entry—can involve examining all the ACI directives for that object. This is because of the hierarchical nature of ACPs and the inheritance of policies from superior ACPs to subordinate ACPs.

The directory server first examines the ACI directives in the entry-level ACI, `orclEntryLevelACI`. It proceeds to the nearest ACP, then considers each superior ACP in succession until the evaluation is complete. This section contains the below topics:

- Attribute States During ACL Evaluation
- Introduction to Precedence Rules Used in ACL Evaluation
- Usage of More Than One ACI for the Same Object
- Exclusionary Access to Directory Objects
- About ACL Evaluation For Groups

31.1.6.1 Attribute States During ACL Evaluation

During ACL evaluation, an attribute is said to be in one of the states described in Table 31-4:

Table 31-4 Attribute States During ACL Evaluation

State	Description
Resolved with permission	The required access for the attribute has been granted in the ACI.
Resolved with denial	The required access for the attribute has been explicitly denied in the ACI.
Unresolved	No applicable ACI has yet been encountered for the attribute in question.

In all operations except search, the evaluation stops if:

- Access to the entry itself is denied
- Any of the attributes reach the resolved with denial state

In this case the operation would fail and the directory server would return an error to the client.

In a search operation, the evaluation continues until all the attributes reach the resolved state. Attributes that are resolved with denial are not returned.

31.1.6.2 Introduction to Precedence Rules Used in ACL Evaluation

An LDAP operation requires the BindDN, or subject, of the LDAP session to have certain permissions to perform operations on the objects—including the entry itself and the individual attributes of the entry.

Typically, there could be a hierarchy of access control administration authorities, starting from the root of a naming context down to successive administrative points (or access control policy points). An ACP is any entry which has a defined value for the `orclACI` attribute. Additionally, the access information specific to a single entry can also be represented within the entry itself (`orclEntryLevelACI`).

ACL evaluation involves determining whether a subject has sufficient permissions to perform an LDAP operation. Typically an `orclentryLevelACI` or `orclACI` might not contain all the necessary information for ACL evaluation. Hence, all available ACL information is processed in a certain order until the evaluation is fully resolved.

That order of processing follows these rules:

- The entry level ACI is examined first. ACIs in the `orclACI` are examined starting with the ACP closest to the target entry and then its superior ACP and so on.
- At any point, if all the necessary permissions have been determined, the evaluation stops; otherwise, the evaluation continues.
- Within a single ACI, if the entity associated with the session DN matches more than one item identified in the *by* clause, the effective access evaluates to:
  - The union of all the granted permissions in the matching by clause items

    ANDed with
  - The union of all the denied permissions in the matching by clause items

31.1.6.2.1 Preceding at the Entry Level

ACIs at the entry level are evaluated in the following order:

1. With a filter. For example:

```
access to entry filter=(cn=p*)

by group1 (browse, add, delete)
```

2. Without a filter. For example:

```
access to entry

by group1 (browse, add, delete)
```

31.1.6.2.2 Preceding at the Attribute Level

At the attribute level, specified ACIs have precedence over unspecified ACIs.

1. ACIs for specified attributes are evaluated in the following order:
   a. Those with a filter. For example:

```
access to attr=(salary) filter=(salary > 10000)
by group1 (read)
```

b. Those without a filter. For example:

```
access to attr=(salary)
by group1 (search, read)
```

2. ACIs for unspecified attributes are evaluated in the following order:

a. With a filter. For example:

```
access to attr=(*) filter (cn=p*)
by group1 (read, write)
```

b. Without a filter. For example:

```
access to attr=(*)
by group1 (read, write)
```

31.1.6.3 Usage of More Than One ACI for the Same Object

Oracle Internet Directory, enables you to define more than one ACI in the ACP of an object. It processes the ACIs associated with that object and stores them as a single ACI in its internal ACP cache. It then applies all the relevant policies in the multiple ACIs specified in the ACP.

The following example of an ACP illustrates how this works.

```
Access to entry by dn="cn=john" (browse,noadd,nodelete)
Access to entry by group="cn=admingroup" (browse,add,nodelete)
Access to entry by dn=".*,c=us" (browse,noadd,nodelete)
```

In this ACP, there are three ACIs for the object entry. When it loads this ACP, Oracle Internet Directory merges these three ACIs as one ACI in its internal ACP cache.

The ACI syntax is:

```
Access to OBJECT> by SUBJECT ACCESSLIST
OBJECT = [ entry | attr [EQ-OR-NEQ] ( * | ATTRLIST ) ]
[ filter = ( LDAFILTER ) ]
```

This syntax makes possible the following types of objects:

- Entry
- Entry + filter = (*LDAPFILTER*)
- Attr = (*ATTRLIST*)
- Attr = (*ATTRLIST*) + filter = (*LDAPFILTER*)
- Attr!= (*ATTRLIST*)
- Attr!= (*ATTRLIST*) + filter = (*LDAPFILTER*)
- Attr = (*)
- Attr = (*) + filter = (*LDAPFILTER*)

You can define multiple ACIs for any of the above types of objects. During initial loading of the ACP, the directory server merges the ACIs based on which of these object types are defined. The matching criterion is the exact string comparison of the object strings in the ACIs.

If one ACI specifies `ATTR=(ATTRLIST)` and another `ATTR!=(ATTRLIST)`, then `ATTR=(*)` must not be specified as an ACI in the entry. Also, if an ACI specifies `ATTR=(ATTRLIST)`, then, to specify the access rights to attributes not in `ATTRLIST`, `ATTR=(*)` must be used and not `ATTR!=(ATTRLIST)`. `ATTR=(*)` implies all attributes other than those specified in `ATTRLIST`.

> **Note:**
>
> When you define multiple ACIs on the same attribute with the same filter, Oracle Internet Directory merges them to create a single ACI in the run-time structure.
>
> When you define multiple ACIs on the same attribute with different filters, Oracle Internet Directory treats them as separate ACIs. In such cases, the precedence order is non-deterministic.
>
> To prevent ambiguous behavior, if you define multiple ACIs with different filters against the same attribute, ensure that the filters yield non-overlapping sets of results.

31.1.6.4 Exclusionary Access to Directory Objects

If an ACI exists for a given object, you can specify access to all other objects except that one. You do this either by granting access to all the objects, or by denying access to the one object.

In the following example, access is granted to all attributes:

```
access to attr=(*)  by group2 (read)
```

In the following example, access is denied to the `userpassword` attribute:

```
access to attr!=(userpassword) by group2 (read)
```

31.1.6.5 About ACL Evaluation For Groups

If an operation on an attribute or the entry itself is explicitly denied at an ACP low in the DIT, then, typically, the ACL evaluation for that object is considered "Resolved with Denial." However, if the user of the session (bindDN) is a member of a group object, then the evaluation continues as if it is still unresolved. If permissions are granted to the user of the session at an ACP higher in the tree through a group subject selector, then such grants have precedence over any denials lower in the DIT.

This scenario is the only case in which an ACL policy at a higher level ACP has precedence over an ACP policy lower in the DIT.

31.2 Managing Access Control by Using Oracle Directory Services Manager

You can view and modify access control information within ACPs by using either Oracle Directory Services Manager or command-line tools.

This section explains how to accomplish these tasks by using Oracle Directory Services Manager.

> **Note:**
>
> Immediately after installing Oracle Internet Directory, be sure to reset the default security configuration.

- Viewing an ACP by Using Oracle Directory Services Manager
- Adding an ACP by Using Oracle Directory Services Manager
- Modifying an ACP by Using Access Control Management in ODSM
- Adding or Modifying an ACP by Using the Data Browser in ODSM
- Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM

> **See Also:**
>
> Oracle Identity Management Command-Line Tool Reference in *Reference for Oracle Identity Management* for a description of command-line tools

31.2.1 Viewing an ACP by Using Oracle Directory Services Manager

You can locate and view an ACP as follows:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Click **Access Control** in the left pane. All of the defined access control points (ACPs) appear in the left pane, listed by relative DN. Mouse over an entry to see the full DN.
4. Select an ACP to display its information in the right pane.
5. The **Subtree Access Items** section of the page shows the access controls on this ACP for entry level operations, that is, for operations on the entry itself.

   The **Content Access Items** section of the page shows the access controls on this ACP for attribute level operations, that is, for operations on the attributes of the entry.

31.2.2 Adding an ACP by Using Oracle Directory Services Manager

ACPs are entries that contain prescriptive, that is, inheritable, access control information. This information affects the entry itself and all entries below it. You will most likely create ACPs to broadcast large-scale access control throughout a subtree.

Adding an ACP by using Oracle Directory Services Manager involves three tasks:

- Specifying the Entry That Will Be the ACP
- Configuring Structural Access Items—that is, ACIs that pertain to *entries*
- Configuring Content Access Items—that is, ACIs that pertain to *attributes*

> **See Also:**
>
> Deleting a Structural or Content Access Item

31.2.2.1 Specifying the Entry That Will Be the ACP

To specify the entry as ACP, perform the following steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Click **Access Control** in the left pane. All of the defined ACPs appear in the left pane.
4. In the left pane, click the **Create Access Control Policy Point** icon. The New Access Control Point screen appears.
5. Enter the path to the entry you want to create, or click **Browse**, select a DN, and click **OK**.

   If the entry is already defined as an ACP, you see an error dialog
6. Alternatively, to create an ACP that is similar to an existing ACP, select the existing ACP in the list under **Access Control Management** in the left pane and click the **Create Like** icon. The New Access Control Point: Create Like screen appears.

31.2.2.2 Configuring Structural Access Items

To configure structural access items, perform the following steps:

1. To define a new structural access item, that is, an ACI that pertains to an entry, choose the **Create new access item** icon in the **Structural Access Items** section of the New Access Control Point pane or the New Access Control Point: Create Like pane. The Structural Access Item dialog box appears.

   Alternatively, you can create a structural access item similar to an existing item. Select an existing structural access item and choose the **Create Copy of Selected Access Item** icon in the **Structural Access Items** section of the he New Access Control Point or New Access Control Point: Create Like pane. The Structural Access Item dialog box appears. Some of the tabs are populated with configuration information, which you can modify as necessary.

   The structural Access Item dialog box has four tabs: **Entry Filter**, **Added Object Filter**, **By Whom**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **Added Object Filter**, **By Whom**, and **Access Rights** tabs to specify the access control.
2. If you want all entries below the ACP to be governed by the ACP, then you do not need to enter anything on the **Entry Filter** tab page; simply proceed to the next step. Otherwise, perform this step.

   You might restrict access to an entry based on one or more of that entry's attributes. For example, you might choose to restrict access to all entries in which the title is manager and in which the organization unit is Americas.

   To use an existing filter, select **Existing** from the Filter Type menu., then select one of the existing filters.

   To create a new filter, select **Create New** from the Filter Type menu, then either type a query string directly into the **LDAP Query** text field, or use the lists and text fields on the search criteria bar to focus your search.

   a. From the list at the left end of the search criteria bar, select an attribute of the entry for which you want to search. Because not all attributes are used in every entry, be sure that the attribute you specify actually corresponds to one in the entry for which you are looking. Otherwise, the search fails.

   b. From the list in the middle of the search criteria bar, select a filter.

   c. In the text box at the right end of the search criteria bar, type the value for the attribute you just selected. For example, if the attribute you selected was `cn`, you could type the particular common name you want to find.

d. Click **+** to add this search criterion to the **LDAP Query** field.

e. To further refine your search, use the list of conjunctions (**AND**, **OR**, **NOT AND**, and **NOT OR**) and the lists and text fields on the search criteria bar to to add additional search criteria. Click **+** to add a search criterion to the **LDAP Query** field.Click **X** to delete a search criterion from the **LDAP Query** field.

3. Select the **Added Object Filter** tab page.

   You can specify ACIs to restrict the kind of entries a user can add. For example, you can specify an ACI in the DSE root entry that allows users to add only entries with `objectclass=country`. The directory server then verifies that any new entry complies with the constraints in this filter.

   To restrict the kind of entries a user can add either type a query string directly into the **LDAP Query** text field, or use the lists and text fields on the search criteria bar to focus your search, following the same steps as in the **Entry Filter** tab page.

4. Select the **By Whom** tab page.

   a. In the By Whom field, specify the entity or entities to whom you are granting access.

   b. From the **Authentication Choice** list under Bind Mode, select the type of authentication to be used by the subject (that is, the entity that seeks access).

      If you do not choose an authentication method, then any kind of authentication is accepted. The authentication method specified on one node should match the one specified on the node it is communicating with.

      From the **Encryption Choice** list under Bind Mode, select the type of encryption to be used.

5. Select the **Access Rights** tab page.

   Specify the kinds of rights to be granted:

   - **Browse**: Allows the subject to see the entry
   - **Add**: Allows the subject to add other entries below this entry
   - **Delete**: Allows the subject to delete the entry
   - **Proxy**: Allows the subject to impersonate another user

6. Click **OK**. The structural access item you just created appears in the list.

Repeat Configuring Structural Access Items to configure additional structural access items in this ACP.

31.2.2.3 Configuring Content Access Items

To configure content access items, perform the following steps:

1. To define content access items, that is, ACIs that pertain to attributes, choose the **Create new access item** icon in the **Content Access Items** section of the New Access Control Point pane or the New Access Control Point: Create Like pane. The Content Access Item dialog box appears.

   Alternatively, you can create a content access item similar to an existing item. Select an existing content access item and choose the **Create copy of selected access item** icon in the **Content Access Items** section of the he New Access Control Point or New Access Control Point: Create Like pane. The Content Access Item dialog box appears. Some of the tabs are populated with configuration information, which you can modify as necessary.

   The Content Access Item dialog box has four tabs: **Entry Filter**, **By Whom**, **Attribute**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to

which an ACI pertains. Use the **By Whom**, **Attribute**, and **Access Rights** tabs to specify the access control.

2. If you want all entries below the ACP to be governed by the ACP, then you do not need to enter anything on **Entry Filter** tab page; simply proceed to the next step. Otherwise, perform this step.

   In an ACP, the access rights apply to the entry and all its subentries unless other filters restrict access further. If appropriate, use the **Entry Filters** tab page to identify the entries to which you are specifying access.

   You might restrict access to an entry based on one or more of that entry's attributes. For example, you might choose to restrict access to all entries in which the title is manager and in which the organization unit is Americas.

   To use an existing filter, select **Existing** from the Filter Type menu, then select one of the existing filters.

   To create a new filter, select **Create New** from the Filter Type menu, then either type a query string directly into the **LDAP Query** text field, or use the lists and text fields on the search criteria bar to focus your search.

   a. From the list at the left end of the search criteria bar, select an attribute of the entry for which you want to search. Because not all attributes are used in every entry, be sure that the attribute you specify actually corresponds to one in the entry for which you are looking. Otherwise, the search fails.

   b. From the list in the middle of the search criteria bar, select a filter.

   c. In the text box at the right end of the search criteria bar, type the value for the attribute you just selected. For example, if the attribute you selected was `cn`, you could type the particular common name you want to find.

   d. Click **+** to add this search criterion to the **LDAP Query** field.

   e. To further refine your search, use the list of conjunctions (**AND**, **OR**, **NOT AND**, and **NOT OR**) and the lists and text fields on the search criteria bar to to add additional search criteria. Click **+** to add a search criterion to the **LDAP Query** field. Click **X** to delete a search criterion from the **LDAP Query** field.

3. Select the **By Whom** tab page.

   a. In the By Whom field, specify the entity or entities to whom you are granting access.

   b. From the **Authentication Choice** list under Bind Mode, select the type of authentication to be used by the subject (that is, the entity that seeks access).

      If you do not choose an authentication method, then any kind of authentication is accepted. The authentication method specified on one node should match the one specified on the node it is communicating with.

      From the **Encryption Choice** list under Bind Mode, select the type of encryption to be used.

   c. Specify the entity or entities to whom you are granting access.

4. Select the **Attribute** tab page.

   a. From Attribute list, select the attribute to which you want to grant or deny access.

   b. From the Operator list, select the matching operation to be performed against the attribute. Choices are EQ (Equal (=)) and NEQ (Not Equal (!=)).

      For example, if you select EQ and `cn`, then the access rights you grant apply to the `cn` attribute. If you select NEQ and `cn`, then the access rights you grant do not apply to the `cn` attribute.

5. Select the **Access Rights** tab page.

   Specify the kinds of rights to be granted or denied:

   - **Read**: Allows the subject to read the attribute
   - **Search**: Allows the subject to search for the attribute
   - **Write**: Allows the subject to change the attribute
   - **Selfwrite**: Allows the subject specified by the entry itself to change the attribute
   - **Compare**: Allows the subject to compare the value of the attribute with that of other attributes.

6. Click **OK**. The content access item you just created appears in the list.

Repeat Configuring Content Access Items to configure additional content access items in this ACP.

31.2.2.4 Deleting a Structural or Content Access Item

To delete a structural or content access item, select the item and click the **Delete** icon.

31.2.3 Modifying an ACP by Using Access Control Management in ODSM

To modify an ACP by using access control management in ODSM, perform the following steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Click **Access Control** in the left pane. All of the defined ACPs appear in the left pane.
4. In the left pane, select an ACP in the list. A tab page for that ACP appears in the right pane.
5. To define a new structural access item, that is, an ACI that pertains to an entry, choose the **Create** icon in the **Structural Access Items** section of the entry tab page. The Structural Access Item dialog box appears.

   Alternatively, you can create a structural access item similar to an existing item. Select an existing structural access item and choose the **Create Like** icon in the **Structural Access Items** section of the he New Access Control Point or New Access Control Point: Create Like pane. The Structural Access Item dialog box appears. Some of the tabs are populated with configuration information, which you can modify as necessary.

   The Structural Access Item dialog box has four tabs: **Entry Filter**, **Added Object Filter**, **By Whom**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **Added Object Filter**, **By Whom**, and **Access Rights** tabs to specify the access control. Follow the steps beginning with Step 2 under Configuring Structural Access Items.

6. To modify an existing structural access item, select the item and click **Edit**. The Structural Access Item dialog box appears. Some of the tabs are populated with configuration information, which you can modify as necessary. The tabs are: **Entry Filter**, **Added Object Filter**, **By Whom**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **Added Object Filter**, **By Whom**, and **Access Rights** tabs to specify the access control. Follow the steps beginning with Step 2 under Configuring Structural Access Items. When you click **OK**, the changes you made to the structural access item are reflected in the list.

7. To define a new content access item, that is, an ACI that pertains to attributes, choose the **Create** icon in the **Content Access Items** section of the entry tab page. The Content Access Item dialog box appears.

   Alternatively, you can create a content access item similar to an existing item. Select an existing content access item and choose the **Create Like** icon in the **Content Access Items** section of the he New Access Control Point or New Access Control Point: Create Like pane. The Content Access Item dialog box appears. Some of the tabs are populated with configuration information, which you can modify as necessary.

   The Content Access Item dialog box has four tabs: **Entry Filter**, **By Whom**, **Attribute**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **By Whom**, **Attribute**, and **Access Rights** tabs to specify the access control. Follow the steps beginning with Step 2 under Configuring Content Access Items.

8. To modify an existing content access item, select the item and click **Edit**. The Content Access Item dialog box appears. It has four tabs: **Entry Filter**, **By Whom**, **Attribute**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **By Whom**, **Attribute**, and **Access Rights** tabs to specify the access control.hey are: **Entry Filter**, **Added Object Filter**, **By Whom**, and **Access Rights**. You can use the **Entry Filters** tab page to limit the subtree entries to which an ACI pertains. Use the **Added Object Filter**, **By Whom**, and **Access Rights** tabs to specify the access control. Follow the steps under Configuring Content Access Items. When you click **OK**, the changes you made appear in the list.

9. To delete a structural or content access item, select the item and click the **Delete** icon.

10. Click **Apply** to effect the changes.

31.2.4 Adding or Modifying an ACP by Using the Data Browser in ODSM

To set subtree-level access by using the Data Browser in Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager as described in Invoking Oracle Directory Services Manager.

2. From the task selection bar, choose **Data Browser**.

3. Navigate to the entry you want to set access to.

4. In the navigator pane, select the entry to display its properties in the right pane

5. Select the **Subtree Access** tab page, then create and edit local ACIs in the **Structural Access Item** and **Content Access Item** tabs as described inModifying an ACP by Using Access Control Management in ODSM.

6. After you have made the changes, click **Apply**.

> **Note:**
>
> You must click **Apply** to send the information you just entered to the directory server. Otherwise, the information is simply held in the Oracle Directory Services Manager cache.

31.2.5 Setting or Modifying Entry-Level Access by Using the Data Browser in ODSM

To set entry-level access by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, choose **Data Browser**.
3. Navigate to the entry you want to set access to.
4. In the navigator pane, select the entry to display its properties in the right pane
5. Select the **Local Access** tab page, then create and edit local ACIs in the **Structural Access Item** and **Content Access Item** tabs as described in Modifying an ACP by Using Access Control Management in ODSM.
6. After you have made the changes, click **Apply**.

> **Note:**
>
> You must click **Apply** to send the information you just entered to the directory server.

31.3 Managing Access Control by Using Command-Line Tools

You can manage access control by using command-line tools, including ldapmodify and ldapmodifymt, to set and alter the values of these attributes.

As described in Overview of Directory Access Control , directory access control policy information is represented as user-modifiable configuration attributes. You can manage it by using command-line tools, including ldapmodify and ldapmodifymt, to set and alter the values of these attributes.

To directly edit the ACI, you should understand the format and semantics of the directory representation of the ACI as described in The Access Control Directive Format . The below topics describe managing access control using command-line tool:

- Restricting the Kind of Entry a User Can Add
- Setting Up an Inheritable ACP by Using ldapmodify
- Setting Up Entry-Level ACIs by Using ldapmodify
- Using Wildcards in an LDIF File with ldapmodify
- Selecting Entries by DN
- Using Attribute and Subject Selectors
- Granting Read-Only Access
- Granting Selfwrite Access to Group Entries
- Defining a Completely Autonomous Policy to Inhibit Overriding Policies

See Also:

- General LDIF Formatting Rules in *Reference for Oracle Identity Management* for information about how to format input by using LDIF, the required input format for line mode commands
- The`ldapmodify` command-line tool reference in *Reference for Oracle Identity Management* for information about how to run ldapmodify
- The Access Control Directive Format for information about the format or syntax of ACI

31.3.1 Restricting the Kind of Entry a User Can Add

You can specify ACIs to restrict the kind of entries a user can add. For example, you can specify an ACI in the DSE root entry that allows users to add only entries with `objectclass=country`.

To do this, you use the `added_object_constraint` filter. The directory server then verifies that any new entry complies with the constraints in this filter.

The following example specifies that:

- The subject `cn=admin,c=us` can browse, add, and delete under `organization` entries.
- The subject `cn=admin,c=us` can add `organizationalUnit` objects under `organization` entries
- All others can browse under `organization` entries

```
access to entry filter=(objectclass=organization)
by group="cn=admin,c=us"
          constraintonaddedobject=(objectclass=organisationalunit)
          (browse,add,delete)
by * (browse)
```

31.3.2 Setting Up an Inheritable ACP by Using ldapmodify

This example sets up subtree access permissions in an `orclACI` at the root DSE by using an LDIF file named `my_ldif_file`.

Because this example refers to the `orclACI` attribute, this access directive governs all the entries in the DIT.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The LDIF file, `my_ldif_file`, contains the following:

```
dn:
changetype: modify
replace: orclaci
orclaci: access to entry

 by dn="cn=directory manager, o=IMC, c=us" (browse, add, delete)
 by * (browse, noadd, nodelete)

orclaci: access to attr=(*)
```

```
 by dn="cn=directory manager, o=IMC, c=us" (search, read, write, compare)
 by self (search, read, write, compare)
 by * (search, read, nowrite, nocompare)
```

31.3.3 Setting Up Entry-Level ACIs by Using ldapmodify

This example sets up entry-level access permissions in the `orclEntryLevelACI` attribute by using an LDIF file named `my_ldif_file`.

Because this example refers to the `orclentrylevelACI` attribute, this access directive governs only the entry in which it resides.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The LDIF file, `my_ldif_file`, contains the following:

```
dn:
changetype: modify
replace: orclentrylevelaci
orclentrylevelaci: access to entry

 by dn="cn=directory manager, o=IMC, c=us" (browse, add, delete)
 by * (browse, noadd, nodelete)

orclentrylevelaci: access to attr=(*)

 by dn="cn=directory manager, o=IMC, c=us" (search, read, write, compare)
 by * (search, read, nowrite, nocompare)
```

> **Note:**
>
> In this example, no DN value is specified. This means that this ACI pertains to the root DSE and its attributes only.

31.3.4 Using Wildcards in an LDIF File with ldapmodify

This example shows the use of wildcards (*) in the object and subject specifiers.

For all entries within the `example.com` domain, it grants to everyone browse permission on all entries, and read and search permissions on all attributes.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

In the ACP at `dc=com`, the `orclACI` attribute is specified as follows:

```
access to entry by * (browse)
access to attr=(*) by * (search, read)
```

Note that, in order to enable reading the attributes, you must grant permission to browse the entries.

31.3.5 Selecting Entries by DN

This example shows the use of a regular expression to select the entries by DN in two access directives.

It grants to everyone read-only access to the address book attributes under `dc=example,dc=com` access.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The `orclACI` attribute of `dc=example,dc=com` is specified as follows:

```
access to entry by * (browse)
access to attr=(cn, telephone, email) by * (search, read)
```

The `orclACI` attribute of `dc=us, dc=example,dc=com` is specified as follows:

```
access to entry by * (browse)
access to attr=(*) by dn=".*,dc=us,dc=example,dc=com" (search, read)
```

31.3.6 Using Attribute and Subject Selectors

This example shows the use of an attribute selector to grant access to a specific attribute, and various subject selectors.

The example applies to entries in the `dc=us,dc=example,dc=com` subtree. The policy enforced by this ACI can be described as follows:

- For all entries within the subtree, the administrator has add, delete, and browse permissions. Others within the `dc=us` subtree can browse, but those outside it have no access to the subtree.
- The salary attribute can be modified by your manager and viewed by yourself. No one else has access to the salary attribute.
- The `userPassword` attribute can be viewed and modified by yourself and the administrator. Others can only compare this attribute.
- The `homePhone` attribute can be read and written by yourself and viewed by anyone else.
- For all other attributes, only the administrator can modify values. Everyone else can compare, search, read, but cannot update attribute values.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The `orclACI` attribute of `dc=us,dc=example,dc=com` is specified as follows:

```
access to entry
by dn="cn=admin, dc=us,dc=example,dc=com" (browse, add, delete)
by dn=".*, dc=us,dc=example,dc=com" (browse)
by * (none)


access to attr=(salary)
by dnattr=(manager) (read, write)
by self (read)
by * (none)


access to attr=(userPassword)
by self (search, read, write)
```

```
by dn="cn=admin, dc=us,dc=example,dc=com" (search, read, write)
by * (compare)


access to attr=(homePhone)
by self (search, read, write)
by * (read)


access to attr != (salary, userPassword, homePhone)
by dn="cn=admin, dc=us,dc=example,dc=com" (compare, search, read, write)
by * (compare, search, read)
```

31.3.7 Granting Read-Only Access

This example gives to everyone read-only access to address book attributes under `dc=example,dc=com`.

It also extends to everyone read access to all attributes within the `dc=us,dc=example,dc=com` subtree only.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The `orclACI` attribute of `dc=example,dc=com` is specified as follows:

```
access to entry by * (browse)
access to attr=(cn, telephone, email) by * (search, read)
```

The `orclACI` attribute of `dc=us,dc=example,dc=com` is specified as follows:

```
access to entry by * (browse)
access to attr=(*) by dn=".*,dc=us,dc=example,dc=com" (search, read)
```

31.3.8 Granting Selfwrite Access to Group Entries

This example enables people within the US domain to add or remove only their own name (DN) to or from the `member` attribute of a particular group entry.

This example enables people within the US domain to add or remove only their own name (DN) to or from the `member` attribute of a particular group entry— for example, a mailing list.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The `orclEntryLevelACI` attribute of the group entry is specified as follows:

```
access to attr=(member)
by dn=".*, dc=us,dc=example,dc=com" (selfwrite)
```

31.3.9 Defining a Completely Autonomous Policy to Inhibit Overriding Policies

This example denies group override.

This example denies group override.

```
ldapmodify -v -h myhost -D "cn=Directory Manager, o=IMC, c=US" -q -f my_ldif_file
```

The example uses the following DNs:

Table 31-5 DNs Used in Example

Container	DN
Naming context to be restricted from Group overriding policies	`c=us`
User container	`cn=users,c=us`
Sensitive data	`cn=appdata`
User admin group for this naming context	`cn= user admin group, cn=users,c=us`
Security admin group or this naming context	`cn= security admin group, cn=users,c=us`
Global password admin group for all naming contexts that reset passwords	`cn=password admin group`

The policy requirements for `c=us` are as follows:

- Users can browse and read their information.
- The user security admin can modify the information under `c=us` except for passwords and ACPs.
- The security admin group can modify policies under `c=us`.
- The global password admin and the user can reset a password.
- All other users have no permissions.
- This policy cannot be overridden.

Required ACP:

```
Access to entry DenyGroupOverride
by dn=".*,c=us" (browse,noadd,nodelete)
by group="cn=User admin group,cn=users,c=us" (browse,add,delete)

Access to attr=(orclaci) DenyGroupOverride
by group="cn=security admin group,cn=users,c=us" (search,read,write,compare)
by * (none)

Access to attr=(userpassword) DenyGroupOverride
by self (search,read,write,compare)
by group="cn=password admin group" (search,read,write,compare)
by * (none)

Access to attr=(*) DenyGroupOverride
by self (search,read,nowrite,compare)
by group="cn= User admin group,cn=users,c=us" (search,read,write,compare)
by * (none)
```

32
Managing Password Verifiers

This chapter describes password verifiers, which are the security credentials used to authenticate users to Oracle Internet Directory and other Oracle components. Specifically, it explains how Oracle Internet Directory centrally stores these password verifiers and how to manage password verifiers using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities.
The following topics are covered:

- Password Verifiers for Directory Authentication
- Managing Hashing Schemes for Password Verifiers for Directory Authentication
- Password Verifiers for Components Authentication
- Managing Password Verifier Profiles for Oracle Components by Using ODSM
- Managing Password Verifier Profiles for Components by Using Command-Line Tools
- Introduction to Verifiers Generation by Using Dynamic Parameters
- Configuring Oracle Internet Directory to Generate Dynamic Password Verifiers

32.1 Password Verifiers for Directory Authentication

When a user leaves a company or changes jobs, that user's privileges should change the same day to guard against misuse of old or unused accounts and privileges.

Without centralized password administration, an administrator in a large enterprise with user accounts and passwords distributed over many databases may not be able to make the changes as quickly as good security requires. The following topics are covered:

- About Oracle Internet Directory
- About Default Password Policy for Oracle Internet Directory
- About Userpassword Verifiers and Directory Authentication
- About Hashing Schemes for Creating Userpassword Verifiers
- Managing Hashing Schemes for Password Verifiers for Directory Authentication

32.1.1 About Oracle Internet Directory

Oracle Internet Directory centrally stores security credentials to make their administration easy for both end users and administrators.

Oracle Internet Directory stores:

- Passwords for authenticating users to the directory itself
- Password verifiers for authenticating users to other Oracle components

Users can store non-Oracle authentication credentials if the non-Oracle applications are directory enabled. These applications must create their own container under the Products entry.

Oracle Internet Directory stores a user's directory password in the `userPassword` attribute. You can protect this password by storing it as a Base64 encoded string of a one-way hashed value by using one of Oracle Internet Directory's supported hashing algorithms. Storing passwords as one-way hashed values—rather than as encrypted values—more fully secures them because a malicious user can neither read nor decrypt them.

32.1.2 About Default Password Policy for Oracle Internet Directory

The default `userPassword` hashing algorithm for Oracle Internet Directory is SSHA512.

All `userPassword` attributes created after a new install are one-way hashed using SSHA512, then stored in Oracle Internet Directory.

When you perform an upgrade, the default hashing scheme is changed to SSHA512. For example, if the default scheme before the upgrade was SSHA, then the default scheme after the upgrade will be SSHA512. Passwords of existing users are not affected. For greater security, require users to reset their passwords, so that SSHA512 hash values are stored in Oracle Internet Directory.

> **Note:**
>
> Support for the MD4/MD5 password hashing algorithm has been removed in 14c (14.1.2.1.0). Therefore, before upgrading, all the `userPassword` must be hashed using at least SSHA. To ensure this, set the default hashing scheme to SSHA and reset the passwords.

Oracle Internet Directory stores the user password in a reversible encrypted format in the `orclrevpwd` configuration attribute. The `orclrevpwd` attribute is generated only if the `orclpwdencryptionenable` attribute in the password policy entry is set to 1.

The `orclrevpwd` attribute is maintained securely within Oracle Internet Directory server and cannot be queried, even if you modify the attribute's access control policy (ACIs). Oracle Directory Integration Platform, however, is allowed to query the `orclrevpwd` attribute, so that password synchronization can function.

32.1.3 About Userpassword Verifiers and Directory Authentication

Oracle Internet Directory can protect a user's directory password by storing it in the `userPassword` attribute as a one-way hashed value.

You select the hashing algorithm you want to use. Storing passwords as one-way hashed values—rather than as encrypted values—more fully secures them because a malicious user can neither read nor decrypt them.

During authentication to a directory server, clients supply a password to the directory server in clear text. The directory server hashes this password by using the hashing algorithm specified in the DSE attribute `orclcryptoscheme`. It then verifies it against the hashed password stored in the binding entry's `userPassword` attribute. If the hashed password values match, then the server authenticates the user. If they do not match, then the server sends the user an "Invalid Credentials" error message.

External users can be authenticated using the `ldapcompare` operation on the binding user's `userpassword` attribute. In this case as well, Oracle Internet Directory uses the same flow as

done during bind operations by first hashing the incoming clear-text value and comparing it against the stored value.

32.1.4 About Hashing Schemes for Creating Userpassword Verifiers

During installation, Oracle Identity Management 14*c* Installer sets the one-way hashing scheme for protecting user passwords to the directory to SSHA. This value is stored in the `orclCryptoScheme` attribute in the root DSE.

You can change that value to any of the following hashing schemes:

- SSHA: Salted Secure Hash Algorithm. This is similar to SHA, but is generated by using a random salt with the password.
- SHA256, SHA384, SHA512: Variants of the more secure SHA-2 algorithm. The numbers refer to the digest sizes of the hash functions.
- SSHA256, SSHA384, SSHA512: Salted versions of the three SHA-2 algorithms.
- None: Passwords are stored in clear text.

See Managing Hashing Schemes for Password Verifiers for Directory Authentication and Managing System Configuration Attributes by Using ODSM Data Browser.

32.2 Managing Hashing Schemes for Password Verifiers for Directory Authentication

The following example changes the password hashing algorithm to SHA512 by using an LDIF file named `my_ldif_file`:

```
ldapmodify -D cn=orcladmin -q -h myhost -p 3060 -v -f my_ldif_file
```

The LDIF file, `my_ldif_file`, contains:

```
dn:
changetype: modify
replace: orclcryptoscheme
orclcryptoscheme: SHA512
```

32.3 Password Verifiers for Components Authentication

Oracle components store both passwords and password verifiers in Oracle Internet Directory.

This section contains these topics:

- About Password Verifiers for Oracle Components Authentication
- Attributes for Storing Password Verifiers for Oracle Components Authentication
- Default Verifiers for Oracle Components
- Understanding Password Verification for an Oracle Component
- Managing Password Verifier Profiles for Oracle Components by Using ODSM
- Managing Password Verifier Profiles for Components by Using Command-Line Tools

32.3.1 About Password Verifiers for Oracle Components Authentication

Oracle components can store their password values in Oracle Internet Directory as password verifiers. A password verifier is a hashed version of a clear text password, which is then encoded as a BASE64 encoded string.

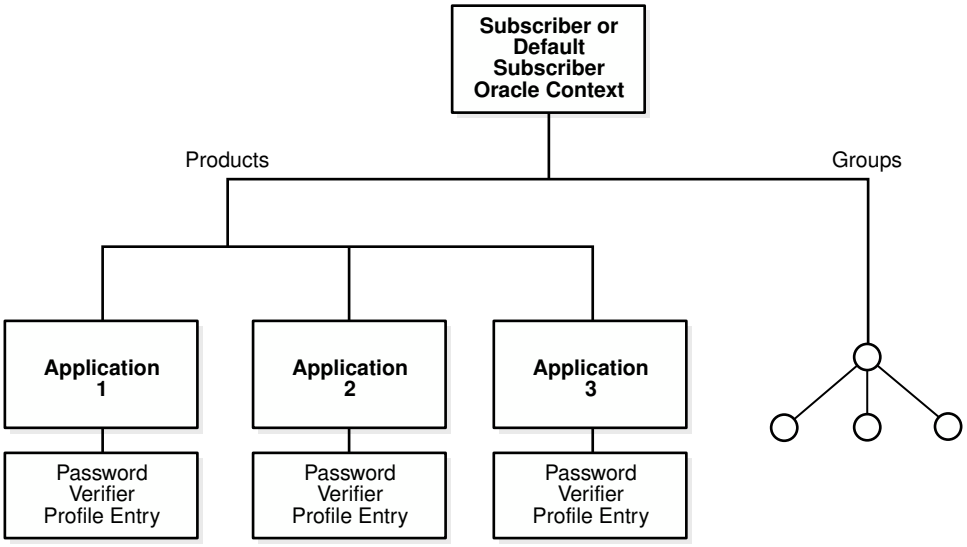
You can choose one of these hashing algorithms to derive a password verifier:

- SSHA
- SHA256, SHA384, SHA512: Variants of the more secure SHA-2 algorithm. The numbers refer to the digest sizes of the hash functions.
- SSHA256, SSHA384, SSHA512: Salted versions of the three SHA-2 algorithms.
- SASL: Simple Authentication and Security Layer, which adds authentication support to connection-based protocols and uses a challenge-response protocol.
- O3LOGON: A proprietary Oracle algorithm for generating verifiers. It is similar to SASL in that it uses a challenge-response protocol.

During Oracle application installation, the Oracle Identity Management 14*c* Installer creates for that application a password verifier profile entry containing all the necessary password verification information. It places this entry, as shown in Figure 32-1, immediately below the application entry, which resides under the products entry, which, in turn, resides under the realm-specific Oracle Context.

This verifier profile entry is applicable to users in the specified realm only. For verifier generation to take effect, you must set the `orclcommonusersearchbase` attribute in the common entry of the realm-specific Oracle context to the appropriate value.

Figure 32-1 Location of the Password Verifier Profile Entry



32.3.2 Attributes for Storing Password Verifiers for Oracle Components Authentication

Both the directory and Oracle components store the user password in the user entry, but in different attributes.

Whereas the directory stores user passwords in the `userPassword` attribute, Oracle components store user password verifiers in the `authPassword`, `orclPasswordVerifier`, or `orclpassword` attribute. Table 32-1 describes each of the attributes used by Oracle components.

Table 32-1 Attributes for Storing Password Verifiers in User Entries

Attribute	Description
`authPassword`	Attribute for storing a password to an Oracle component when that password is the same as that used to authenticate the user to the directory, namely, `userpassword`. The value in this attribute is synchronized with that in the `userpassword` attribute. Several different applications can require the user to enter the same clear text password used for the directory, but each application may hash it with a different algorithm. In this case, the same clear text password can become the source of several different password verifiers. This attribute is multivalued and can contain all the other verifiers that different applications use for this user's clear text password. If the `userpassword` attribute is modified, then the `authpasswords` for all applications are regenerated.
`orclPasswordVerifier`	Attribute for storing a password to an Oracle component when that password is different from that used to authenticate the user to the directory, namely, `userpassword`. The value in this attribute is not synchronized with that in the `userpassword` attribute. Like `authPassword`, this attribute is multivalued and can contain all the other verifiers that different applications use for this user's clear text password.
`orclPassword`	Attribute for storing only the 03LOGON verifier for enterprise users. The 03LOGON verifier is synchronized with the `userpassword` attribute, and it is generated by default for all user entries associated with the `orcluserv2` object class. When Oracle Internet Directory is installed, a database security profile entry is created by default in the Root Oracle Context. The presence of this entry triggers the generation of 03LOGON verifiers for user entries associated with the `orcluserv2` object class.

Each of these attribute types has `appID` as an attribute subtype. This attribute subtype uniquely identifies a particular application. For example, the `appID` can be the `ORCLGUID` of the application entry. This attribute subtype is generated during application installation.

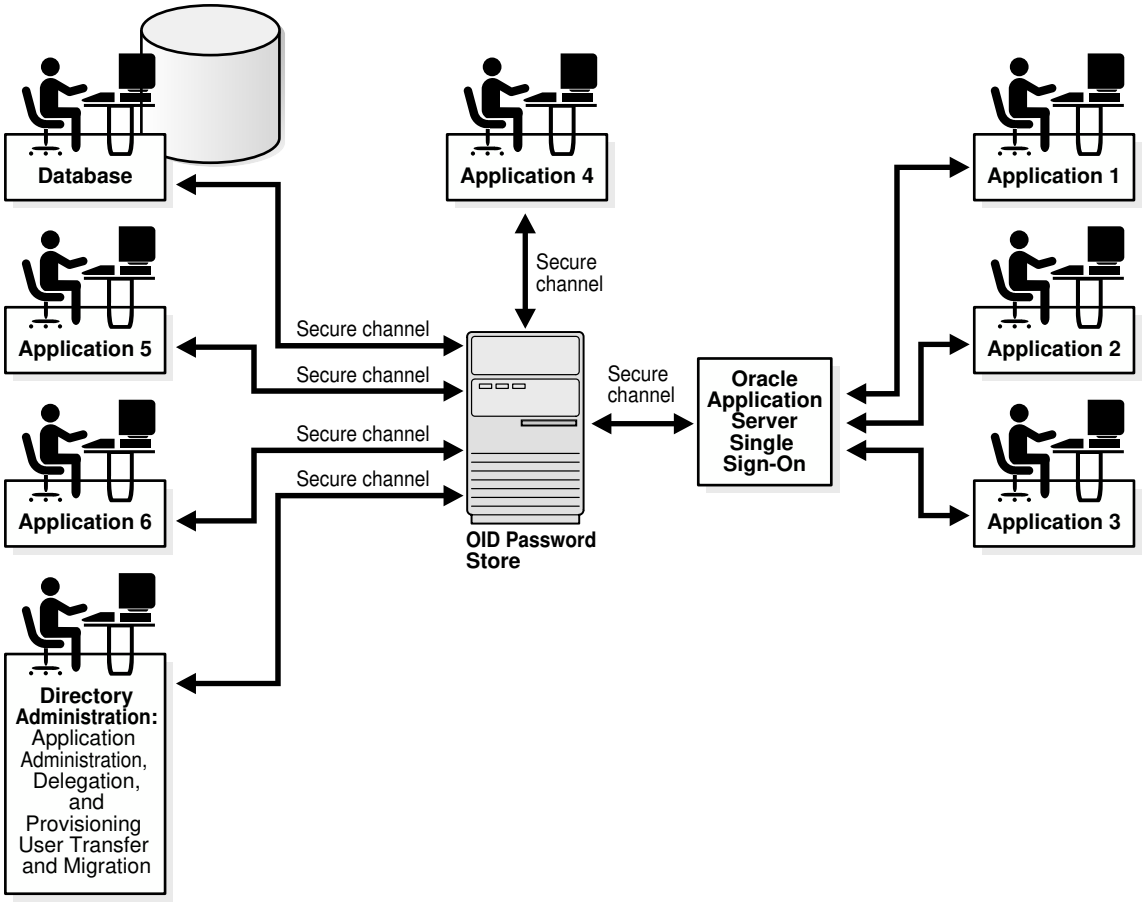
In Figure 32-2, various Oracle components store their password verifiers in Oracle Internet Directory. Oracle Single Sign-On uses the same password as that for the directory, and hence stores it in the `authPassword` attribute. The other applications use different passwords and hence store their verifiers in `orclPasswordVerifier` attribute.

The following is an example of an application-specific verifier profile entry. Any application that chooses not to use the common verifier framework must create its own verifier profile entry, similar to the one given in the following example. The `orclappid` is set to the GUID of the application container and it will also be used as a subtype in the verifier attributes `authpassword` and `orclpasswordverifier`.

```
dn: cn=IFSVerifierProfileEntry,cn=IFS,cn=Products,cn=OracleContext,o=Oracle,dc=com
objectclass:top
objectclass:orclpwdverifierprofile
cn:IFSVerifierProfileEntry
orclappid:8FF2DFD8203519C0E034080020C34C50
orclpwdverifierparams;authpassword: crypto:SASL/MD5 $ realm:dc=com
orclpwdverifierparams;orclpasswordverifier: crypto:SSHA512
```

SASL/MD5 verifiers are generated by using user name, realm, and password. The user name attribute to be used can be specified in the verifier profile entry. The case of the user name can also be specified as either upper or lower.

Figure 32-2 Authentication Model



32.3.3 Default Verifiers for Oracle Components

To save you from having to create a profile for each Oracle component, and to enable sharing of password verifiers across all components, Oracle Internet Directory provides a default set of password verifiers.

The default verifier type is SSHA512.

Two profile entries are required: one for applications using personal identification numbers (PINs), which use numeric values only, and another for applications using alphanumeric passwords.

The verifiers for PIN-based applications—for example, the voice mail application in OracleAS Unified Messaging—are stored in the `orclpasswordverifier;orclcommonpin` attribute. The subtype `orclcommonpin` is used to distinguish numeric PINs from alphanumeric passwords. Any application that uses numeric PINs can directly query or compare against the attribute `orclpasswordverifier;orclcommonpin`.

The verifiers for alphanumeric password-based applications can be stored in either:

- The `authpassword;orclcommonpwd` attribute—If an application requires its verifier to be synchronized with the `userpassword` attribute
- The `orclpasswordverifier;orclcommonpwd` attribute—If synchronization with the `userpassword` attribute is not required

The subtype `orclcommonpwd` is used to distinguish alphanumeric passwords from numeric PINs. The verifier attributes subtyped by orclcommonpwd can be queried against.

These profile entries also contain the list of subscribed applications and these are specified as values in the `uniquemember` attribute in the profile entries. By default, the DN of the Oracle Single Sign-On identity is one of the subscribed applications. This means that Oracle Single Sign-On is a proxy member for all its partner applications. All applications not based on Oracle Single Sign-On must add their identities (DNs) to the `uniquemember` attribute in the appropriate profile entry.

The following is an example of the profile entries.

```
Cn=defaultSharedPwdProfileEntry, cn=common, cn=products, cn=oraclecontext
Objectclass: orclpwdverifierprofile
Objectclass: orclcommonpwdprofile
Cn: defaultSharedPwdProfileEntry
Orclappid: orclcommonpwd
Orclpwdverifierparams;authpassword: crypto:SASL/MD5 $ realm:Authorized_Users
Orclpwdverifierparams;orclpasswordverifier: crypto:SSHA
Uniquemember: cn=SSO,cn=Products,cn=OracleContext
Uniquemember: cn=IFS,cn=Products,cn=OracleContext

Cn=defaultSharedPINProfileEntry, cn=common, cn=products, cn=oraclecontext
Objectclass: orclpwdverifierprofile
Objectclass: orclcommonpwdprofile
Cn: defaultSharedPinProfileEntry
Orclappid: orclcommonpin
Orclpwdverifierparams;orclpasswordverifier: crypto:SSHA
Uniquemember: cn=SSO,cn=Products,cn=OracleContext
Uniquemember: cn=Unified Messaging,cn=Products,cn=OracleContext
```

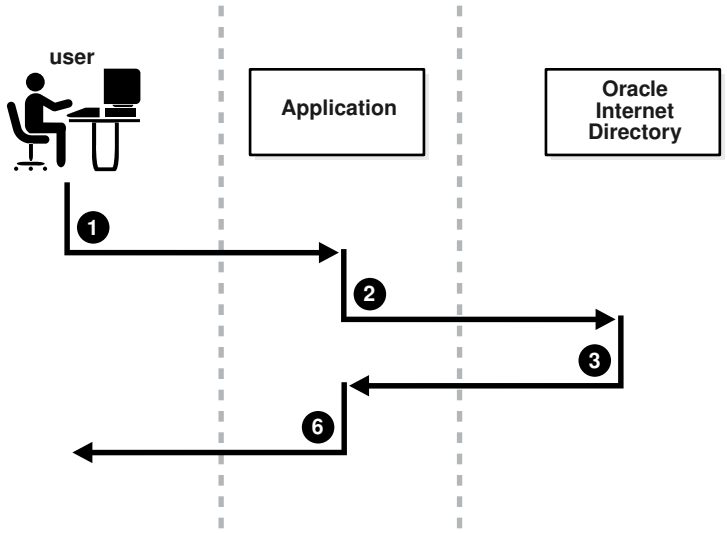
For PIN-based applications, `authpassword` is not an option. Such applications use the `orclpasswordverifier` attribute.

32.3.4 Understanding Password Verification for an Oracle Component

In this example, the Oracle component stores its password verifiers in the directory.

Figure 32-3 shows an example of password verification for an Oracle component. In this example, the Oracle component stores its password verifiers in the directory.

Figure 32-3 How Password Verification Works



1. The user tries to log in to an application by entering a user name and a clear text password.
2. The application sends the clear text password to the directory server. If the application stores password verifiers in the directory, then the application requests the directory server to compare this password value with the corresponding one in the directory.
3. The directory server:
   a. Generates a password verifier by using the hashing algorithm specified for the particular application
   b. Compares this password verifier with the corresponding password verifiers in the directory. For the compare operation to be successful, the application must provide its `appID` as the subtype of the verifier attribute. For example:

```
ldapcompare -p 3060 -D "DN_of_the_appplication_entity" -q \
            -b "DN_of_the_user" -a orclpasswordverifier;appID \
            -v password_of_the_user
```

   c. Notifies the application of the results of the compare operation.
4. Depending on the message from the directory server, the application either authenticates the user or not.

If an application does not use the compare operation, then it:

- Hashes the clear text password entered by the user
- Retrieves from the directory the hashed value of the clear text password as entered by the user

- Initiates a challenge to the user to which the client responds. If the response is correct, then the application authenticates the user.

32.4 Managing Password Verifier Profiles for Oracle Components by Using ODSM

You can use Oracle Directory Services Manager to view and modify password verifier profile entries, including password hashing schemes.

To view and modify an application's password verifiers:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Security**.
3. Expand **Password Verifier** in the left pane. All of the password verifiers appear in the left pane.
4. Select the password verifier you want to view. The right pane displays the Password Verifier Profile tab page.
5. You can modify the hashing algorithm used to generate a password verifier. In the Password Verifier Profile dialog box, specify the hashing algorithm in the **Oracle Password Parameters** field. The syntax is:

   ```
   crypto:hashing_algorithm
   ```

   For example, if you are using the SSHA512 hashing algorithm, then you would enter:

   ```
   crypto:SSHA512
   ```

   If you are using SASL/MD5, for example, you can enter the following:

   ```
   crypto:SASL/MD5 $ realm:dc=com
   ```

32.5 Managing Password Verifier Profiles for Components by Using Command-Line Tools

You can view and modify password verifier profiles by using command-line tools.

This is described in the following sections:

- Viewing a Password Verifier Profile by Using Command-Line Tools
- Modifying a Password Verifier Profile by Using Command-Line Tools

32.5.1 Viewing a Password Verifier Profile by Using Command-Line Tools

To view an application's password verifier, perform a search specifying the DN of the password verifier profile.

To view an application's password verifier, perform a search specifying the DN of the password verifier profile.

32.5.2 Modifying a Password Verifier Profile by Using Command-Line Tools

This example changes the hashing algorithm in an application password verifier profile entry.

This password verifier synchronizes with the user's directory password. Type:

```
ldapmodify -D "cn=orcladmin" -q -p 3060 -h my_host -v -f file.ldif
```

where `file.ldif` contains:

```
dn: cn=MyAppVerifierProfileEntry,cn=MyApp,cn=Products,cn=OracleContext,
 o=my_company,dc=com
changetype: modify
replace: orclPwdVerifierParams
orclPwdVerifierParams;authPassword: crypto:SASL/MD5 $ realm:dc=com
```

32.6 Introduction to Verifiers Generation by Using Dynamic Parameters

The password verifiers described previously are static password verifiers. That is, they are generated with preconfigured parameters, typically during application installation.

Some applications, including Oracle Calendar, Oracle Email, and Oracle Wireless and Voice, require Oracle Internet Directory to generate dynamic password verifiers.

Oracle Internet Directory generates a dynamic password verifier when an application requests one. Dynamic verifiers are based on application parameters that are not available until run time.

In order to generate a dynamic password verifier, Oracle Internet Directory needs a user password that was previously stored in a reversible encrypted format. Oracle Internet Directory stores such values in the configuration attributes `orclrevpwd` and `orclunsyncrevpwd`. Encrypted values based on `userpassword` are stored in the parameter `orclrevpwd`. Encrypted values based on passwords other than `userpassword`, such as the numeric PINs used by Oracle Calendar, are stored in the parameter `orclunsyncrevpwd`.

32.7 Configuring Oracle Internet Directory to Generate Dynamic Password Verifiers

If you are deploying applications that use `userpassword` and that need dynamic password verifiers, you must ensure that Oracle Internet Directory generates the `orclrevpwd` attribute.

Oracle Internet Directory generates the attribute `orclrevpwd` when you provision a user if the attribute `orclpwdencryptionenable` in the realm password policy entry is set to `1`. Therefore, you must set `orclpwdencryptionenable`to `1` before you provision users.

To check the value of `orclpwdencryptionenable`, you can use `ldapsearch` to query the `pwdpolicy` entry effective on the entry being examined. For example, to query the default `pwdpolicy`, you would type:

```
$ ldapsearch -p port -q -D "cn=orcladmin" \
   -b "cn=default,cn=pwdPolicies,cn=Common,cn=Products,cn=OracleContext" \
   -s base "objectclass=*"
```

If `orclpwdencryptionenable` is not listed or is equal to `0`, set it to `1` by using ldapmodify with an LDIF file similar to this:

```
dn: cn=DefaultSharedPinProfileEntry,cn=Common,cn=Products,cn=Oraclecontext
cn: DefaultSharedPinProfileEntry
orclappid: orclpwdencryptionenable
orclpwdencryptionenable: 1
```

Alternatively, if users were provisioned before you set `orclpwdencryptionenable`, all users must reset their user passwords to trigger the generation of the encrypted value.

If you are deploying applications that use a numeric PIN and that need dynamic password verifiers, you must ensure that Oracle Internet Directory can use the crypto type `3DES` in order to generate the value stored in `orclunsyncrevpwd`. You must specify `3DES` as a value of the attribute `orclpwdverifierparams;orclpasswordverifier` in the common verifier profile entry under the root oracle context. The default DN of this entry is `cn=DefaultSharedPINProfileEntry,cn=Common,cn=Products, cn=OracleContext`. To set the value, you would use the following LDIF file:

```
dn: cn=DefaultSharedPinProfileEntry,cn=Common,cn=Products,cn=Oraclecontext
cn: DefaultSharedPinProfileEntry
orclappid: orclcommonpin
orclpwdverifierparams;orclpasswordverifier: crypto:3DES
```

Use a command-line such as:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

33

Delegating Privileges for Oracle Identity Management

This chapter describes the delegation model for Oracle Identity Management and the default security configuration in Oracle Internet Directory. Specifically, it explains how to store all the data for users, groups, and services in one repository and then how to delegate the administration of that data to various administrators.
This chapter includes the following sections:

- Oracle Identity Management Privileges
- User and Group Management Privileges
- Privileges for Deployment of Oracle Components
- Delegating Privileges for Component Run Time

> **Note:**
>
> All references to Oracle Delegated Administration Services in this chapter refer to Oracle Delegated Administration Services 10g (10.1.4.3.0) or later.

33.1 Oracle Identity Management Privileges

Oracle Identity Management enables you to store all the data for users, groups, and services in one repository, and to delegate a particular administrator for each set of data. By providing both a centralized repository and customized delegated access, Oracle Identity Management is both secure and scalable.

This section contains these topics:

- About Privileges Delegation
- About Delegation in an Oracle Fusion Middleware Environment
- Default Configuration
- Privileges for Administering the Oracle Technology Stack

33.1.1 About Privileges Delegation

Using the delegation model, a global administrator can delegate to realm administrators the privileges to create and manage the identity management realms for hosted companies. Realm administrators can, in turn, delegate to end users and groups the privileges to change their application passwords, personal data, and preferences. Each type of user can thus be given the appropriate level of privileges.

To delegate the necessary privileges, you assign the user to the appropriate administrative group. For example, suppose that you store data for both enterprise users and the e-mail service in the directory, and must specify a unique administrator for each set of data. To specify

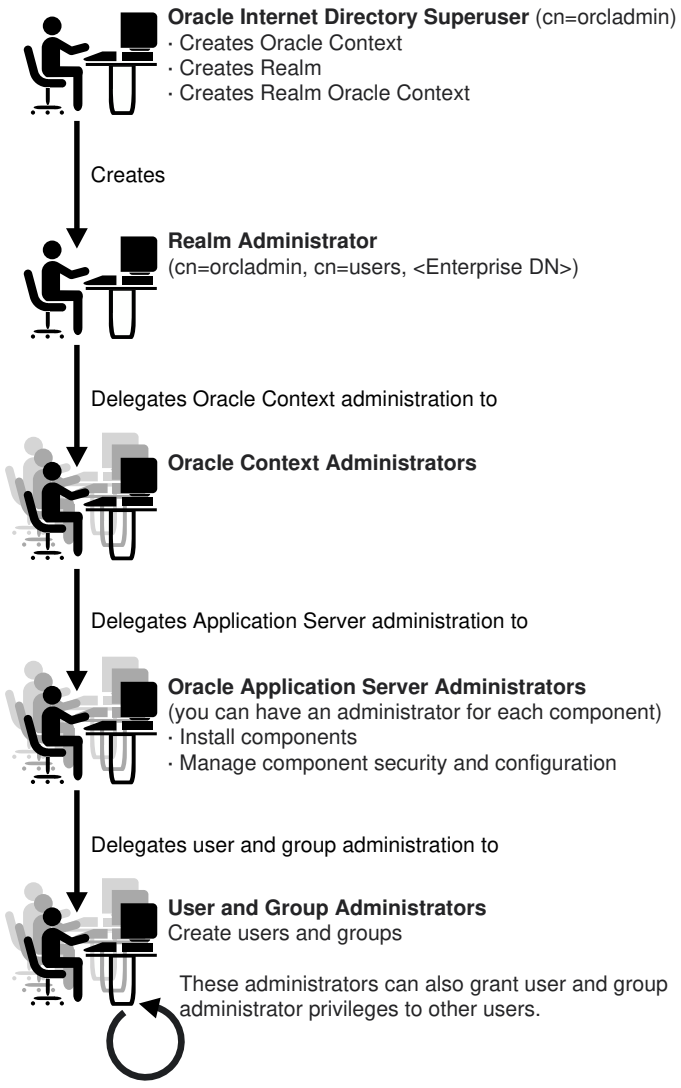
a user as the administrator of enterprise users, you assign that user to, say, the Enterprise User Administrators Group. To specify a user as the administrator of the e-mail services, you assign that user to, say, the E-mail Service Administrators Group.

33.1.2 About Delegation in an Oracle Fusion Middleware Environment

This section introduces you to the flow of delegation in an Oracle Fusion Middleware environment.

Figure 33-1 shows the flow of delegation in an Oracle Fusion Middleware environment.

Figure 33-1 Delegation Flow in an Oracle Fusion Middleware Environment



As Figure 33-1 shows, in an Oracle Fusion Middleware environment the directory superuser (`cn=orcadmin`) creates:

- The Oracle Context
- The realm

- The realm-specific Oracle Context
- The entry for the realm administrator (`cn=orcladmin, cn=users, Enterprise DN`)

The realm administrator, in turn, delegates administration of the Oracle Context to specific users by assigning those users to the Oracle Context Administrators Group. Oracle Context Administrators then delegate administration of the Oracle Application Server to one or more users by assigning them to the Oracle Fusion Middleware Administrators Group. The Oracle Fusion Middleware Administrators install and administer Oracle Fusion Middleware components and delegate administration of user and group data to the User and Group Administrators group. The User and Group Administrators create users and groups. They can also grant user and group administrator privileges to other users.

33.1.3 Default Configuration

When you first install Oracle Internet Directory, the default configuration establishes access control policies at various points in the directory information tree (DIT). Default access controls are placed on the User and Group containers as described later in this chapter. Likewise, default privileges for specific directory entities are discussed later in this chapter.

In addition, certain default privileges are granted to Everyone and to each user as described in Table 33-2.

Table 33-1 Default Privileges Granted to Everyone and to Each User

Subject	Default Privileges
Everyone	The following privileges at the Root DSE: • Permission to browse user entries • Search, read, and compare access for all user attributes except the following userpkcs12, orcluserpkcs12hint, userpassword, orclpassword, and orclpasswordverifier
Each user	Complete access to his or her own attributes—including the `userpassword`, `orclpassword`, and `orclpasswordverifier` attributes.

You can customize this default configuration to meet the security requirements of your enterprise.

33.1.4 Privileges for Administering the Oracle Technology Stack

This section describes the types of privileges and points to additional information about each one.

Administering the Oracle technology stack requires the privileges described in Table 33-2.

Table 33-2 Privileges for Administering the Oracle Technology Stack

Type of Privilege	Description	More Information
User and group management privileges	These are delegated to either Oracle components that use the identity management infrastructure or to end users themselves	User and Group Management Privileges

Table 33-2 (Cont.) Privileges for Administering the Oracle Technology Stack

Type of Privilege	Description	More Information
Deployment-time privileges	These are required to deploy any Oracle component. They may include privileges to create appropriate entries inside the directory, or to store metadata in a common repository. Such privileges must be given, for example, to an administrator of Oracle Portal.	Privileges for Deployment of Oracle Components
Run-time privileges	These are required to facilitate the run-time interactions of Oracle components within the identity management infrastructure. These include privileges to view user attributes, add new users, and modify the group membership. Such privileges must be given to the administration tool specific to each Oracle component, enabling it to access or create entries inside Oracle Internet Directory.	Delegating Privileges for Component Run Time

> **Note:**
>
> Be careful when modifying the default ACLs in any Oracle Context. Doing so can disable the security of Oracle components in your environment. See component-specific documentation for details on whether you can safely modify the default ACLs in an Oracle Context.

> **See Also:**
>
> Understanding Data Migration from Other Data Repositories if you have an existing directory structure that you now want to migrate to an Oracle Application Server environment

33.2 User and Group Management Privileges

Administrative privileges are delegated to either Oracle components that use the identity management infrastructure or to end users themselves.

A privilege can be delegated to either an identity—for example, a user or application—or to a role or group. The following sections describe user and group management privileges in detail:

- About Privileges for Managing User and Group Data
- Default Privileges for Managing User Data
- Default Privileges for Managing Group Data

33.2.1 About Privileges for Managing User and Group Data

This procedure helps you to delegate administrative privileges as a super user.

To delegate administrative privileges, the Oracle Internet Directory superuser does the following:

1. Creates an identity management realm
2. Identifies a special user in that realm who is called the realm administrator
3. Delegates all privileges to that realm administrator

This realm administrator, in turn, delegates certain privileges that Oracle components require to the Oracle defined roles—for example, Oracle Fusion Middleware administrators. The Oracle components receive these roles when they are deployed.

In addition to delegating privileges to roles specific to Oracle components, the realm administrator can also define roles specific to the deployment—for example, a role for help desk administrators—and grant privileges to those roles. These delegated administrators can, in turn, grant these roles to end users. In fact, because a majority of user management tasks involve self-service—like changing a phone number or specifying application-specific preferences—these privileges can be delegated to end users by both the realm administrator and Oracle component administrators.

In the case of a group, one or more owners—typically end users—can be identified. If they are granted the necessary administrative privileges, then these owners can manage the group by using Oracle Internet Directory Self-Service Console, Oracle Directory Services Manager, or command-line tools.

33.2.2 Default Privileges for Managing User Data

The access control policy point (ACP) for creating users is at the Users container in the identity management realm.

These sections describe each of these privileges in more detail

- Privileges for Creating Users for a Realm
- Privileges for Modifying Attributes of a User
- Privileges for Deleting a User
- Privileges for Delegating User Administration

33.2.2.1 Privileges for Creating Users for a Realm

To create users for a realm, an administrator must be a member of the Subscriber DAS Create User Group. Table 33-3 describes the characteristics of this group.

Table 33-3 Characteristics of the Subscriber DAS Create User Group

Characteristic	Description
Default ACP	The ACL at the Users container in the default realm allows the Subscriber DAS Create User Group in the realm Oracle Context to create users under the Users container.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the User Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASCreateUser,cn=Groups,`*`Oracle_Context_DN`*.

33.2.2.2 Privileges for Modifying Attributes of a User

To modify user attributes, an administrator must be a member of the Subscriber DAS Edit User Group. Table 33-4 describes the characteristics of this group.

Table 33-4 Characteristics of the Subscriber DAS Edit User Group

Characteristic	Description
Default ACP	The ACL at the Users container in the default identity management realm allows the Subscriber DAS Edit User Group in the realm Oracle Context to modify various attributes of users.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the User Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASEditUser,cn=Groups,`*`Oracle_Context_DN`*

33.2.2.3 Privileges for Deleting a User

To delete a user in a realm, an administrator must be a member of the DAS Delete User Group. Table 33-5 describes the characteristics of this group.

Table 33-5 Characteristics of the DAS Delete User Group

Characteristic	Description
Default ACP	The ACL at the Users container in the default identity management realm allows the DAS Delete User Group in the realm Oracle Context to delete a user from the realm.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the User Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASDeleteUser,cn=Groups,`*`Oracle_Context_DN`*

33.2.2.4 Privileges for Delegating User Administration

A delegated administrator can perform specified operations within the directory and requires permission to add any user to the User Creation, User Edit, or User Delete Groups described previously.

To grant user administration privileges to a delegate administrator, the granting administrator must be a member of the User Privilege Assignment Group. Table 33-6 describes the characteristics of this group.

Table 33-6 Characteristics of the User Privilege Assignment Group

Characteristic	Description
Default ACP	The ACL policy for each of the groups previously mentioned allows members of the User Privilege Assignment Group to add users to or remove them from those groups.
Administrators	The Oracle Internet Directory superuser Oracle Context Administrators Group Owners of this group. The DNs of these owners are listed as values of the `owner` attribute in the group.
DN	`cn=OracleDASUserPriv,cn=Groups,`*`Oracle_Context_DN`*

33.2.3 Default Privileges for Managing Group Data

This section helps you understand the default privileges for managing group data.

Managing users and groups involves privileges to:

- Create and delete group entries
- Modify group attributes
- Delegate group administration to other users

The ACP for creating groups is at the Groups container in the identity management realm. The details are described in the below sections:

- Privileges for Creating Groups
- Privileges for Modifying the Attributes of Groups
- Privileges for Deleting Groups
- Privileges for Delegating Group Administration

33.2.3.1 Privileges for Creating Groups

To create groups in Oracle Internet Directory, an administrator must be a member of the Group Creation Group. Table 33-7 describes the characteristics of this group.

Table 33-7 Characteristics of the Group Creation Group

Characteristic	Description
Default ACP	The ACL at the Groups container in the realm allows the Group Creation Group to add new groups in the realm.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the Oracle Fusion Middleware Administrators Group Members of the Group Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASCreateGroup,cn=Groups,`*`Oracle_Context_DN`*

33.2.3.2 Privileges for Modifying the Attributes of Groups

To modify the attributes of groups under the Groups container in a realm, an administrator must be a member of the Group Edit Group. Table 33-8 describes the characteristics of this group.

Table 33-8 Characteristics of the Group Edit Group

Characteristic	Description
Default ACP	The ACL at the Groups container in the realm allows the Group Edit Group to modify various attributes of groups in the realm.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the Oracle Fusion Middleware Administrators Group Members of Group Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASEditGroup,cn=Groups,`*`Oracle_Context_DN`*

33.2.3.3 Privileges for Deleting Groups

To delete groups, an administrator must have membership in the Group Delete Group. Table 33-9 describes the characteristics of this group.

Table 33-9 Characteristics of the Group Delete Group

Characteristic	Description
Default ACP	The ACL at the Groups container in the realm allows the Group Delete Group to delete groups in the realm.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the Group Privilege Assignment Group Members of the DAS Administrators Group Owners of this group
DN	`cn=OracleDASDeleteGroup,cn=Groups,`*`Oracle_Context_DN`*

33.2.3.4 Privileges for Delegating Group Administration

To delegate group administration to other users—that is, to add or remove users from the Group Creation, Group Edit, or Group Delete Groups described previously—an administrator must be a member of the Group Privilege Assignment Group. Table 33-10 describes the characteristics of this group.

Table 33-10 Characteristics of the Group Privilege Assignment Group

Characteristic	Description
Default ACP	The ACL policy for the Group Creation, Group Edit, or Group Delete Groups allows members of Group Privilege Assignment Group to add users to or remove them from those groups.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Owners of the group. The DNs of these owners are listed as values of the `owner` attribute in the group.
DN	`cn=OracleDASUserPriv,cn=Groups,`*`Oracle_Context_DN`*

33.3 Privileges for Deployment of Oracle Components

This section discusses the groups responsible for deploying Oracle components. It describes the tasks these administrators perform and the privileges they can grant.

Refer to the following topics:

- Granting Deployment Privileges
- Oracle Application Server Administrators
- User Management Application Administrators
- Trusted Application Administrators

> **Note:**
>
> Oracle Internet Directory superusers have all the privileges of Oracle Fusion Middleware Administrators and Trusted Application administrators, and must be members of the Oracle Fusion Middleware Administrators Group. They can:
>
> – Assign the Oracle Fusion Middleware Administrator role to a user
>
> – Assign the Trusted Application role to a user
>
> – Assign the User Management Application Administrator role to a user

33.3.1 Granting Deployment Privileges

This section describes how to grant deployment privileges.

To enable administrators to deploy Oracle components, the superuser:

1. Grants certain deployment privileges to various groups—for example, the Oracle Fusion Middleware Administrators Group
2. Adds the administrators to those privileged groups

The delegated administrators, in turn, can delegate privileges to other administrators.

33.3.2 Oracle Application Server Administrators

This section describes the characteristics of the Oracle Application Server Administrators group.

Table 33-11 describes the characteristics of the Oracle Application Server Administrators Group.

Table 33-11 Characteristics of the Oracle Application Server Administrators Group

Characteristic	Description
Tasks	Perform repository database installation that creates a repository database registration entry in the directory Perform mid-tier installation. To associate a mid-tier with a repository, the user must have the appropriate privileges with a specific repository database. Install and configure Oracle Fusion Middleware components that create application entities in Oracle Internet Directory Grant to component entities the run-time privileges listed later in this section Configure provisioning profiles for components so that the components can receive update notifications
Privileges this group can delegate to components	Read Common User Attributes—except passwords, certificates, and similar security credentials Read common group attributes Create, edit, and delete groups Authenticate a user Read application verifiers
Administrators	Oracle Internet Directory superuser Oracle Context Administrator Owners of this group
DN	`cn=iASAdmins,cn=Groups,`*`Oracle_Context_DN`*

33.3.3 User Management Application Administrators

User Management Application Administrators must be members of the Oracle Fusion Middleware Administrators Group.

Table 33-12 describes the characteristics of the User Management Application Administrators Group.

Table 33-12 Characteristics of the User Management Application Administrators Group

Characteristic	Description
Tasks	User Management Application administrators install specific applications that have interfaces to perform user management operations—for example, Oracle Portal and Oracle Application Server Wireless.
Privileges this group can delegate to components	Create, edit, and delete user attributes

Table 33-12 (Cont.) Characteristics of the User Management Application Administrators Group

Characteristic	Description
Administrators	Oracle Internet Directory superuser Oracle Context Administrator Owners of this group
DN	`cn=IAS&UserMgmtApplicationAdmins,cn=Groups,`*`Oracle_Context_DN`

33.3.4 Trusted Application Administrators

Trusted Application administrators must be members of the Oracle Fusion Middleware Administrators Group.

Table 33-13 describes the characteristics of the Trusted Application Administrators Group.

Table 33-13 Characteristics of the Trusted Application Administrators Group

Characteristic	Description
Tasks	Install specific identity management components—for example, Oracle Single Sign-On, Oracle Identity Management, and Oracle Application Server Certificate Authority
Privileges this group can delegate to components	Read, compare, or reset the user password Proxy as the end-user Read, compare, or modify the user's certificate and SMIME certificate
Administrators	Oracle Internet Directory superuser Oracle Context Administrator Owners of this group
DN	`cn=TrustedApplicationsAdmins,cn=Groups,`*`Oracle_Context_DN`

33.4 Delegating Privileges for Component Run Time

Many Oracle components administer user entries in Oracle Internet Directory and need the corresponding privileges.

This section describes the security privileges required by Oracle components. It contains these topics:

- Authenticating Oracle Single Sign-On Server
- Default Privileges for Reading and Modifying User Passwords
- Default Privileges for Comparing User Passwords
- Default Privileges for Comparing Password Verifiers
- Default Privileges for Proxying on Behalf of End Users
- Default Privileges for Managing the Oracle Context
- Default Privileges for Reading Common User Attributes
- Default Privileges for Reading Common Group Attributes

- Default Privileges for Reading the Service Registry
- Default Privileges for Administering the Service Registry

33.4.1 Authenticating Oracle Single Sign-On Server

This section describes how to authenticate Oracle single sign-on server.

- When the Oracle Single Sign-On server authenticates a user, that server:
  - Connects to Oracle Internet Directory using its own identity
  - Verifies that the password entered by the user matches that user's password stored in the directory

  To do this, the Oracle Single Sign-On server needs permission to compare user passwords. To set up the Oracle Single Sign-On cookie, it needs permission to read user attributes.
- To grant access to a user, Oracle Portal must retrieve that user's attributes. To do this, it logs in to Oracle Internet Directory as a proxy user, impersonating the user seeking access. It therefore needs the privileges of a proxy user.

In general, Oracle components require these privileges:

- Read and modify user passwords
- Compare user passwords
- Proxy on behalf of users accessing applications
- Administer the Oracle Context where all Oracle components store their metadata

Most Oracle components ship with a preconfigured set of privileges. You can change these default privileges to satisfy specific business requirements—for example, by removing privileges to create and delete user entries.

> **See Also:**
>
> *Oracle® Application Server Security Guide* in the 14c (14.1.2.1.0) library for further information about the component delegation model.

33.4.2 Default Privileges for Reading and Modifying User Passwords

Reading and modifying user passwords requires administrative privileges on the security-related attributes in the directory.

For example, the `userPassword` attribute. It requires membership in the User Security Administrators Group described in Table 33-14.

Table 33-14 Characteristics of the User Security Administrators Group

Characteristic	Description
Default ACP	The default ACL policy at the Root (DSE Entry) allows members of the User Security Administrators Group to read, write, compare, and search on `userpkcs12`, `orclpkcs12hint`, `userpassword`, `orclpassword`, and `orclpasswordverifier` attributes at the Root Oracle Context. However, directory administrators can grant similar administrative privileges to the User Security Administrators Group in the realm Oracle Context.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the Trusted Application Administrators Group
DN	`cn=OracleUserSecurityAdmins,cn=Groups,`*`Oracle_Context_DN`*

33.4.3 Default Privileges for Comparing User Passwords

Comparing user passwords requires permission to compare a user's `userPassword` attribute. This operation is performed by components such as Oracle Unified Messaging that authenticate end users by using their passwords stored in Oracle Internet Directory.

Comparing user passwords requires membership in the Authentication Services Group described in Table 33-15.

Table 33-15 Characteristics of the Authentication Services Group

Characteristic	Description
Default ACP	The ACL policy at the Users container in the default identity management realm allows the Authentication Services Group to perform compare operation on the `userPassword` attribute of users.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Members of the Application Server Administrators Group Owners of this group
DN	`cn=authenticationServices,cn=Groups,`*`Oracle_Context_DN`*

33.4.4 Default Privileges for Comparing Password Verifiers

To compare password verifiers, a user must have permission to compare the `userpassword` attribute.

Comparing password verifiers requires membership in the Verifier Services Group described in Table 33-16.

Table 33-16 Characteristics of the Verifier Services Group

Characteristic	Description
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators group Members of the Application Server Administrators group Owners of this group
DN	`cn=verifierServices,cn=Groups,`*`Oracle_Context_DN`*

33.4.5 Default Privileges for Proxying on Behalf of End Users

A proxy user has the privilege to impersonate an end user, performing on that user's behalf those operations for which that user has privileges. In such a case, the access controls on the directory server eventually govern the operations that the user can perform.

Proxying on behalf of end users requires membership in the User Proxy Privilege Group described in Table 33-17.

Table 33-17 Characteristics of the User Proxy Privilege Group

Characteristic	Description
Default ACP	The ACL at the Users container in the default identity management realm allows User Proxy Privilege Group to proxy on behalf of the end user.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group Owners of the groups. The DNs of these owners are listed as values of the `owner` attribute in the group or members of the Oracle Fusion Middleware Administrators Group. Members of the Trusted Application Administrators Group
DN	`cn=UserProxyPrivilege,cn=Groups,`*`OracleContextDN`*

33.4.6 Default Privileges for Managing the Oracle Context

To manage a specific Oracle Context, a user must have complete access to it.

Managing an Oracle Context requires membership in the Oracle Context Administrators Group described in Table 33-18. An Oracle Context Administrators Group exists for each Oracle Context and has administrative permission in the specific Oracle Context.

Table 33-18 Characteristics of the Oracle Context Administrators Group

Characteristic	Description
Default ACP	The ACL policy at the root node of the Oracle Context allows members of Oracle Context Administrators Group to perform all administrative operations within the Oracle Context. Such a policy is set up when a new Oracle Context is created in the directory.
Administrators	The Oracle Internet Directory superuser Members of the Oracle Context Administrators Group
DN	`cn=OracleContextAdmins,cn=Groups,`*`Oracle_Context_DN`*

33.4.7 Default Privileges for Reading Common User Attributes

Common user attributes are: `mail`, `orclguid`, `displayname`, `preferredlanguage`, `orcltime`, `gender`, `dateofbirth`, `telephonenumber`, `wirelessaccountnumber`.

To read these attributes requires membership in the Common User Attributes Group described in Table 33-19.

Table 33-19 Characteristics of the Common User Attributes Group

Characteristic	Description
Default ACP	The default ACL is on the User container in the realm and grants permission to read common user attributes.
Administrators	The Oracle Internet Directory superuser Members of the Application Server Administrators Group Owners of this group
DN	`cn=CommonUserAttributes,cn=Groups,`*`Oracle_Context_DN`*

33.4.8 Default Privileges for Reading Common Group Attributes

Common group attributes are: `cn`, `uniquemember`, `displayname`, and `description`. To read these attributes requires membership in the Common Group Attributes Group.

Common Group Attributes Group is described in Table 33-20.

Table 33-20 Characteristics of the Common Group Attributes Group

Characteristic	Description
Default ACP	The default ACL is on the Group container in the realm and grants permission to read these attributes: `cn`, `uniquemember`, `displayname`, and `description`.
Administrators	The Oracle Internet Directory superuser Members of the Application Server Administrators Group Owners of this group
DN	`cn=CommonGroupAttributes,cn=Groups,`*`Oracle_Context_DN`*

33.4.9 Default Privileges for Reading the Service Registry

To view the contents of the Service Registry requires membership in the Service Registry Viewers Group.

Service Registry Viewers Group is described in Table 33-21.

Table 33-21 Characteristics of the Service Registry Viewers Group

Characteristic	Description
Default ACP	The default ACL is on the Services container in the root Oracle Context.

Table 33-21 (Cont.) Characteristics of the Service Registry Viewers Group

Characteristic	Description
Administrators	The Oracle Internet Directory superuser Members of the Application Server Administrators Group Owners of this group
DN	`cn=Service Registry Viewers,cn=Groups,`*`cn=OracleContext`*

33.4.10 Default Privileges for Administering the Service Registry

To administer the Service Registry requires membership in the Service Registry Administrators Group.

Service Registry Administrators Group is described in Table 33-22.

Table 33-22 Characteristics of the Common Group Attributes Group

Characteristic	Description
Default ACP	The default ACL is on the Services container in the root Oracle Context.
Administrators	The Oracle Internet Directory superuser Members of the Application Server Administrators Group Owners of this group
DN	`cn=Service Registry Admins,cn=Groups,`*`cn=OracleContext`*

34
Managing Authentication

This chapter provides an overview of Oracle Internet Directory authentication features, including direct, indirect, and external authentication, the Simple Authentication and Security Layer (SASL), and anonymous binds. Specifically, it describes how to manage authentication using Oracle Enterprise Manager Fusion Middleware Control and LDAP command-line tools. This chapter includes the following sections:

- Introduction to Authentication
- About Certificate Authentication Method by Using Fusion Middleware Control
- Configuring SASL Authentication by Using Oracle Enterprise Manager Fusion Middleware Control
- Configuring Certificate Authentication Method by Using Command-Line Tools
- SASL Authentication by Using the Command Line
- Anonymous Binds
- Managing Anonymous Binds
- Restricting Users from Binding to Oracle Internet Directory Server
- Managing Unauthenticated Binds

34.1 Introduction to Authentication

Authentication is the process by which the directory server establishes the true identity of the user connecting to the directory.

It occurs when an LDAP session is established with the bind operation. Thus every session has an associated user identity.

To verify the identities of users, hosts, and clients, Oracle Internet Directory enables three general kinds of authentication, and these are described in these topics:

- About Direct Authentication
- About Indirect Authentication
- About External Authentication
- Simple Authentication and Security Layer (SASL)

34.1.1 About Direct Authentication

This section describes the three kinds of direct authentication available within Oracle Internet Directory, and about how SASL-enabled clients authenticate to a directory server.

The three kinds of direct authentication options are:

Table 34-1 Direct Authentication Options

Authentication Option	Description
Anonymous authentication	When users authenticate anonymously, they simply leave the user name and password fields blank when they log in. Each anonymous user then exercises whatever privileges are specified for anonymous users.
Simple authentication	When using simple authentication, the client identifies itself to the server with a DN and a password that are not encrypted when sent over the network.
Authentication by using Simple Authentication and Security Layer (SASL)	This is a method for adding authentication support to connection-based protocols. To use SASL, a protocol includes a command for identifying and authenticating a user to a server and for optionally negotiating protection of subsequent protocol interactions. If the use of SASL is successfully negotiated, then a security layer is inserted between the protocol and the connection.

- Oracle Internet Directory supports two authentication mechanisms with SASL:
  - Digest MD5: The required authentication mechanism within LDAP Version 3 (RFC 2829). It uses the MD5 hash function to convert a message of any length to a 128 bit message digest that can be used as a verifier for client/server authentication.
  - External authentication: Mechanism using SSL mutual authentication. In this case, the client, in lieu of a user name and password, authenticates to the server with a certificate, token, or some other device. Certificate authentication can take the following forms:
    - Exact match: the subject DN in the client certificate is compared with the user DN in the directory. If the two values match, a bind occurs.
    - Certificate hash: The client certificate is hashed and is then compared with the hashed value of the certificate stored in the directory. If the two values match and only one DN is associated with the pair, a bind occurs. If two or more DNs are associated, an error is returned because certificate hash and user DN is an n-to-1 mapping and not a 1-to-n mapping. That is, you can have many certificates associated with one DN, but only one DN associated with a certificate.
    - Exact match/certificate hash: An exact-match search is performed first. If this search yields nothing, a certificate hash is performed.

    To choose one of these methods, edit the DSA configuration attribute `orclpkimatchingrule` as prescribed in Oracle Identity Management LDAP Attribute Reference in *Reference for Oracle Identity Management*. (For authentication, an `orclpkimatchingrule` value of 3 or 4 is equivalent to a value of 2.)

Note:

- The introduction of a certificate hash value requires that user certificates be upgraded from earlier releases. To learn how to upgrade certificates, see the `upgradecert.pl` command-line tool reference in *Reference for Oracle Identity Management*.
- You can search for the binary attribute `usercertificate`. To learn how to conduct a search, see Searching the Directory for User Certificates.
- The `usercertificate` attribute is intended to contain DER format certificates only.

See Also:

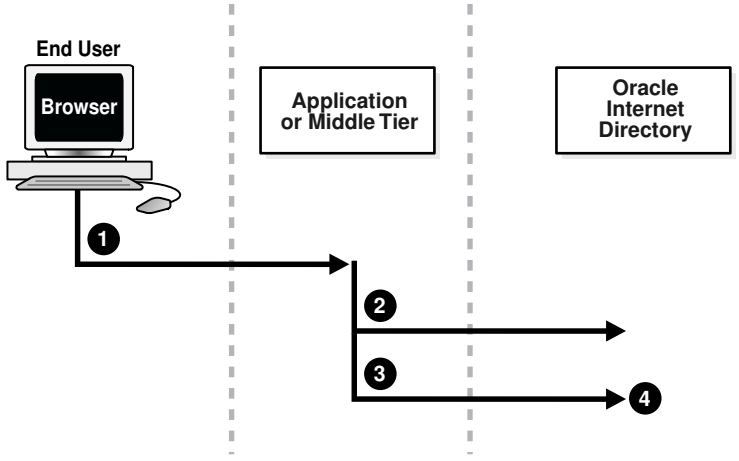
- Simple Authentication and Security Layer (SASL)
- The Web site of the Internet Engineering Task Force (IETF) at `http://www.ietf.org` for the following RFCs: RFC 2829, which specifies SASL Digest-MD5 as the required authentication mechanism for LDAP Version 3 servers; RFC 2831, which describes the Digest-MD5 mechanism; RFC 2617, which describes the HTTP Digest authentication mechanism on which SASL Digest-MD5 is based

34.1.2 About Indirect Authentication

Indirect authentication occurs through any entity that has credentials in the directory—for example, an application such as the Oracle Internet Directory Self-Service Console, or a middle tier such as a firewall or a RADIUS server. The application or middle tier becomes a proxy user. A proxy user has the privilege to impersonate an end user, performing on that user's behalf those operations for which that user has privileges.

Figure 34-1 and the accompanying text explain how indirect authentication takes place.

Figure 34-1 Indirect Authentication



Indirect authentication takes place as follows:

1. The end user sends to the application or middle tier a request containing a query to Oracle Internet Directory. The application or middle tier authenticates the end user.
2. The application or middle tier binds to the directory.
3. The application or middle tier performs a second bind, this time using the DN of the end user. It does not enter the end user's password.
4. The directory server recognizes this second bind as an attempt by the application or middle tier to switch to the end user's identity. It trusts the authentication granted to the end user by the application or middle tier, but must verify that the application or middle tier has the right to be the proxy for this user. It checks to see whether the ACP governing the end user entry gives this application or middle tier the proxy right for this end user.
   - If the end user entry does give the application or middle tier the necessary proxy right, then the directory server changes the authorization identity to that of the end user. All subsequent operations occur as if that end user had connected directly to the server and had been directly authenticated.
   - If the end user entry does not give the application or middle tier the necessary proxy right, then the directory server returns an "Insufficient Access" error message.

> **See Also:**
>
> Access Control to Operations.

The directory server can, in the same session, authenticate and authorize other end users. It can also switch the session from the end user to the application or middle tier that opened the session.

To close the session, the application or middle tier sends an unbind request to the directory server.

For example, suppose you have:

- A middle tier that binds to the directory as `cn=User1`, which has proxy access on the entire directory
- An end user that can bind to the directory as `cn=User2`

When this end user sends to the application or middle tier a request containing a query to the directory, the application or middle tier authenticates the end user. The middle tier service then binds to the directory by using its own identity, `cn=User1`, then performs a second bind, this time by using only the DN of the end user, `cn=User2`. The Oracle directory server recognizes this second bind as an attempt by the proxy user to impersonate the end user. After the directory server verifies that `cn=user1` has proxy access, it allows this second bind to succeed. It does not require any further validation of the end-user DN, such as a password. For the rest of the session, all LDAP operations are access-controlled as if `cn=User2` were performing them.

If one user is being serviced by an application, and another user subsequently requests a service of that same application, then the application can establish a new connection and proceed as previously described without disrupting that prior session. If, however, no prior user is still being serviced, then the existing established connection can be re-used again and again without the need for a new connection.

34.1.3 About External Authentication

Perhaps your enterprise stores user security credentials in a repository other than Oracle Internet Directory—for example, a database or another LDAP directory.

With Oracle Internet Directory external authentication and password modification plug-ins, you can use these credentials for user authentication to Oracle components. You do not need to store the credentials in Oracle Internet Directory and then worry about keeping them synchronized.

> **See Also:**
>
> Configuring a Customized External Authentication Plug-in

34.1.4 Simple Authentication and Security Layer (SASL)

This section describes more fully how SASL works.

The section About Direct Authentication introduced the use of SASL within an Oracle Internet Directory environment. This section describes more fully how SASL works. It contains these topics:

- Authenticating a SASL-Enabled Client to a Directory Server by Using Digest-MD5
- Authenticating a SASL-Enabled Client to a Directory Server by Using External Authentication

34.1.4.1 Authenticating a SASL-Enabled Client to a Directory Server by Using Digest-MD5

> **Note:**
>
> To use SASL Digest-MD5 with realms, SASL must generate a Dynamic User Verifier. You must enable reversible password generation. For more information, see Working with SASL Authentication by Using Digest-MD5 in *Application Developer's Guide for Oracle Identity Management*.

When a SASL-enabled client seeks Digest-MD5 authentication to a server, the authentication process is as follows:

1. The directory server sends to the LDAP client a digest-challenge that includes various Digest-MD5 authentication options that it supports and a special token.
2. The client selects an authentication option, then sends a digest-response to the server indicating the option it has selected. The response includes some secure tokens and a client credential in encrypted format. This allows it to authenticate itself to the server.
3. The directory server then decrypts and verifies the client credential from the response.

34.1.4.2 Authenticating a SASL-Enabled Client to a Directory Server by Using External Authentication

Oracle Internet Directory provides SASL-external authentication over an SSL connection in which both client and server authenticate themselves to each other by providing certificates. The DN is derived from the client certificate used in the SSL network negotiation.

When a client seeks authentication to a directory server by using an external authentication mechanism such as SSL, the authentication process is as follows:

1. The client sends an initial message with the authorization identity.
2. The directory server uses information external to SASL to determine whether the client can validly authenticate as the authorization identity. If the client can validly authenticate, then the directory server indicates successful completion of the authentication exchange. Otherwise, the directory server indicates failure.

The system providing the external information may be IPsec or SSL/TLS. The authorization identity is derived as follows:

- In case of exact match, the authorization identity is derived from the client authentication credentials in the system providing external authentication—for example, the client SSL certificate.
- If the client sends an empty string as the authorization identity, then the authorization identity is derived from the client authentication credentials in the system providing external authentication—for example, the SSL certificate.

34.2 About Certificate Authentication Method by Using Fusion Middleware Control

This section describes certificate authentication method by using Oracle Fusion Middleware Control.

To configure the value of the DSA configuration attribute `orclPKIMatchingRule`, which determines the method used for certificate authentication, use the Oracle Internet Directory **Shared Properties** page of Oracle Enterprise Manager Fusion Middleware Control. Select **Administration**, then **Shared Properties** from the **Oracle Internet Directory** menu. Select the desired matching method from the list for PKI Matching Rule.

34.3 Configuring SASL Authentication by Using Oracle Enterprise Manager Fusion Middleware Control

This section explains the procedure to configure SASL authentication by using Oracle Enterprise Manager Fusion Middleware Control.

To configure SASL Authentication by using Oracle Enterprise Manager Fusion Middleware Control:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu, then select **SASL**.
2. Select the desired types for MD5 SASL Authentication Mode.

3. If you select Authentication with Integrity and Privacy Protection, you are presented with choices for SASL Cipher Choice for Privacy Protection. Select the desired types.

   Choices are `rc4-56`, `des`, `3des`, `rc4`, and `rc4-40`. All five are enabled by default. At least `des` and `3des` must be configured, or SASL authentication fails.

4. If desired, select **Enable SASL Authentication**. Before enabling SASL Authentication, ensure that Oracle Internet Directory is configured to perform mutual authentication. See Overview of Configuring SSL by Using Fusion Middleware Control.

5. Choose **Apply**.

   Restart the server after changing any of the attributes on the SASL tab, as shown in Table 34-2.

Table 34-2 Configuration Attributes on Server Properties, SASL Tab

Field or Heading	Configuration Attribute
MD5 SASL Authentication Mode	`orclsaslauthenticationmode`
SASL Cipher Choice for Privacy Protection	`orclsaslcipherchoice`
External SASL Authentication Mode	`orclsaslmechanism`

34.4 Configuring Certificate Authentication Method by Using Command-Line Tools

To configure the value of the DSA configuration attribute `orclPKIMatchingRule`, which determines the method used for certificate authentication, use `ldapmodify`.

In the command line, type the following command:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

and with an LDIF file similar to this:

```
dn: cn=dsaconfig, cn=configsets, cn=oracle internet directory
changetype: modify
replace: orclPKIMatchingRule
orclPKIMatchingRule: 1
```

The values are:

- 0 - Exact match. The PKI certificate DN must match the user entry DN.
- 1 - Certificate search. Check to see if the user has a PKI certificate provisioned into Oracle Internet Directory.
- 2 - A combination of exact match and certificate search. If the exact match fails, then a certificate search is performed.
- 3 - Mapping rule only. Use a mapping rule to map user PKI certificate DNs to Oracle Internet Directory DNs.
- 4 - Try in order: 1 (mapping rule), 2 (certificate search), 3 (exact match).

34.5 SASL Authentication by Using the Command Line

This section lists and describes the SASL authentication attributes.

Table 34-3 lists the SASL authentication attributes. They reside in the instance-specific configuration entry.

Table 34-3 SASL Authentication Attributes

Attribute	Description	Default	Possible Values
`orclsaslauthenticationmode`	SASL Authentication Mode	1	auth, auth-int, auth-conf.
`orclsaslcipherchoice`	SASL Cipher Choice	Rc4-56,rc4-40,rc4,des,3des	Any combination of Rc4-56, des, 3des, rc4, rc4-40
`orclsaslmechanism`	SASL Mechanism	DIGEST-MD5, EXTERNAL	DIGEST-MD5, EXTERNAL

MD5 SASL Authentication Mode is controlled by the attribute `orclsaslauthenticationmode`. Table 34-4 lists the possible modes. You can specify all three values or any subset as a comma separated string.

Table 34-4 SASL Authentication Modes

orclsaslauthentication mode Value	Mode	Description
1	auth	Authentication only.
2	auth-int	Authentication with integrity protection. A checksum is performed on the channel.
3	auth-conf	Authentication with integrity protection and encryption. The channel is encrypted.

If you set `orclsaslauthenticationmode` to `3` (`auth-conf`), you must also choose values for `orclsaslcipherchoice`. Choices are: `rc4-56`, `des`, `3des`, `rc4`, and `rc4-40`. All five are enabled by default. At least `des` and `3des` must be configured, or SASL authentication fails.

The variable `orclsaslmechanism` controls which authentication mechanisms are supported with SASL. The values `DIGEST-MD5` and `EXTERNAL` are set by default. Do not disable Digest-MD5 authentication. It is the required authentication mechanism for LDAP Version 3 servers. You can disable external authentication by using `ldapmodify`. Disable external authentication if you do not configure Oracle Internet Directory to perform SSL client and server authentication.

Restart the server after changing any of the SASL attributes.

34.6 Anonymous Binds

An anonymous bind is one that uses simple authentication with no password. By default, the directory server allows anonymous bind, but allows only search operations on root DSE entry for anonymous users.

You can configure the server to allow all anonymous binds or to disallow anonymous binds. This behavior is controlled by the `orclanonymousbindsflag` attribute of the s server instance-specific configuration entry. Table 34-5 lists the allowed values for `orclanonymousbindsflag` and the resulting directory server behavior.

Table 34-5 Orclanonymousbindsflag Value and Directory Server Behavior

orclAnonymousBindsFlag Value	Directory Server Behavior
0	Disallows anonymous bind
1	Allows anonymous bind
2	Allows anonymous bind but allows only search operations on root DSE entry for anonymous users (default)

34.7 Managing Anonymous Binds

Anonymous binds are allowed by default, but anonymous users can only perform search operations on the root DSE entry.

You can use either Fusion Middleware Control or the command line to change the server's behavior with respect to anonymous binds. This section contains the following topics:

- Managing Anonymous Binds by Using Fusion Middleware Control
- Managing Anonymous Binds by Using the Command Line

34.7.1 Managing Anonymous Binds by Using Fusion Middleware Control

This section explains the procedure to manage anonymous binds by using Oracle Enterprise Manager Fusion Middleware Control.

Perform the following steps:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu, then select the **General**. tab.
2. From the **Anonymous Binds** list, select **Allows** to enable anonymous binds. Select **Disallow except for Read Access on the root DSE** to allow only search operations on root DSE entry for anonymous users.

To disable anonymous binds by using:

1. Select **Administration**, then **Server Properties** from the **Oracle Internet Directory** menu, then select the **General**. tab.
2. From the **Anonymous Binds** list, select **Disallow**.

34.7.2 Managing Anonymous Binds by Using the Command Line

You can enable all anonymous bind on the Oracle Internet Directory instance with componentName `oid1` using `ldapmodify`.

In the command line, type:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

with an LDIF file such as:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclAnonymousBindsFlag
orclAnonymousBindsFlag: 1
```

To disable all anonymous binds, you would use a similar LDIF file with the last line changed to:

```
orclAnonymousBindsFlag: 0
```

> **See Also:**
>
> Attributes of the Instance-Specific Configuration Entry

34.8 Restricting Users from Binding to Oracle Internet Directory Server

The `bindAuthPriv` attribute allows you to specify the users who can bind to Oracle Internet Directory server.

To specify the users who can bind to the server:

1. Create a directory group that contains only the users who are allowed to bind to the server.
2. Set the `bindAuthPriv` attribute to point to the group you created in Step 1.
3. Add the `bindAuthPriv` attribute to the user entry of each user who is allowed to bind to the server (that is, the users who are members of the group from Step 1).

If a user is not a member of the group, the server rejects subsequent bind requests by that user.

Several other considerations are:

- The `bindAuthPriv` attribute can be a collective attribute that allows specific users to inherit the attribute.
- The directory group can be a nested group.
- You must ensure the appropriate access control (ACLs) for the `bindAuthPriv` attribute, so that only an administrator can add it to a user entry.

> **See Also:**
>
> For more information about creating and managing users and groups using Oracle Directory Services Manager (ODSM) or the LDAP command-line tools:
>
> - Managing Directory Entries in Oracle Internet Directory
> - Managing Dynamic and Static Groups in Oracle Internet Directory

34.9 Managing Unauthenticated Binds

Oracle Internet Directory provides support for the unauthenticated authentication mechanism by sending a Bind request with a distinguished name value (user DN) but no password.

When unauthenticated binds are allowed, the bind attempt goes through as an anonymous bind. This section contains the following topics:

- Configuring Unauthenticated Binds
- Managing Unauthenticated Binds by Using the Command Line

34.9.1 Configuring Unauthenticated Binds

The `orclunauthenticatedbindsflag` attribute allows you to set an unauthenticated bind to succeed.

The `orclunauthenticatedbindsflag` attribute allows you to set an unauthenticated bind to succeed. Setting this attribute to `1` allows unauthenticated bind to succeed. By default, this parameter is set to `0`.

To configure unauthenticated bind, see Managing Unauthenticated Binds by Using the Command Line.

34.9.2 Managing Unauthenticated Binds by Using the Command Line

You can enable unauthenticated bind on the Oracle Internet Directory instance with componentName `oid1` using `ldapmodify`.

In the command line, type:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

with an LDIF file such as:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclunauthenticatedbindsflag
orclunauthenticatedbindsflag: 1
```

To disable unauthenticated bind, you would use a similar LDIF file with the last line changed to:

```
orclunauthenticatedbindsflag: 0
```

Part IV

Advanced Administration: Managing Directory Deployment

This part discusses important deployment considerations.

This chapter includes the following sections:

- Planning, Deploying and Managing Realms
- Tuning and Sizing Oracle Internet Directory
- Managing Garbage Collection
- Migrating Data from Other Data Repositories
- Configuring Directory Server Chaining

35
Planning, Deploying and Managing Realms

The following topics describe the identity management realm implementation in Oracle Internet Directory and how to plan, deploy, and manage identity management realms for enterprise and hosted deployments:

- Understanding Identity Management Realms
- Customizing the Default Identity Management Realm
- About Additional Identity Management Realms for Hosted Deployments
- Creating a DIT View

> **Note:**
>
> All references to Oracle Single Sign-On in this guide refer to Oracle Single Sign-On 10g (10.1.4.3.0) or later.

35.1 Understanding Identity Management Realms

This section gives an overview of planning identity management realms.

This introduction includes the following topics:

- Overview of Identity Management Realm Planning
- Understanding Identity Management Realms in an Enterprise Deployment
- Overview of Identity Management Realms in a Hosted Deployment
- Identity Management Realm Objects
- Overview of Default Identity Management Realm

35.1.1 Overview of Identity Management Realm Planning

This section describes guidelines for you to structure the overall DIT and the placement of users and groups for your deployment. Because implementing these guidelines can lead to an infinite number of deployment configurations, you must capture the intent of your deployment in metadata in the directory itself. This metadata enables Oracle software and other third-party software relying on the Oracle Identity Management infrastructure to understand the deployment intent and successfully function in customized environments

See Understanding Oracle Internet Directory Organization .

In Oracle Internet Directory, this deployment intent is captured in the identity management realm. The realm also helps set identity management policies for users and groups whose placement is described in the previous section.

The identity management realm is a well-scoped area in the directory that consists of:

- A well-scoped collection of enterprise identities—for example, all employees in the US

- A collection of identity management policies associated with these identities
- A collection of groups—that is, aggregations of identities—that makes it easier to set identity management policies

When you have decided on the overall DIT structure and the placement of users and groups, you must identify the directory entry to serve as the root of the identity management realm. This entry determines the scope of the identity management policies defined in the realm. By default, the scope is the entire directory subtree under the root of the identity management realm. Under this entry, a special entry called `OracleContext` is created. It contains the following:

- The deployment-specific DIT design, including user and group naming and placement, as described in previous sections
- The identity management policies associated with this realm
- Additional realm-specific information specific to Oracle applications

When planning the identity management realm, consider the following:

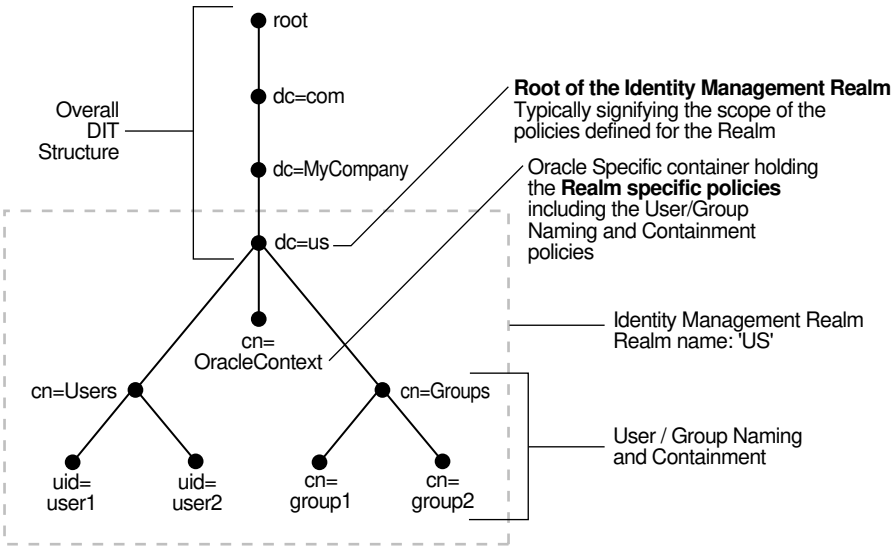
- The security needs of your enterprise must dictate the choice of the root of the identity management realm. Typically, most enterprises need only one realm. However, multiple realms may be required when multiple user populations are managed with different identity management policies.
- If you already have a third-party directory, or plan to integrate with one in the future, then align the choice of the identity management realm root with the DIT design of the third-party directory. This simplifies the synchronization and subsequent administration of the distributed directories.
- To configure and administer identity management realms, use the administrative tools provided by Oracle Internet Directory. These include the Oracle Internet Directory Self-Service Console in Oracle Identity Management 10g (10.1.4.3.0) or later, and command-line tools.

> **See Also:**
>
> The command reference for `oidrealm` in Oracle Internet Directory Realm Tool section in *Reference for Oracle Identity Management*.

- After you have used the Oracle Internet Directory tools to configure the identity management realm, plan on updating the directory naming and containment policies to reflect the customizations made by the deployment. This update must happen before installing and using other Oracle components that use the Oracle Identity Management infrastructure.

Figure 35-1 shows an example of an identity management realm for an enterprise called MyCompany.

Figure 35-1 Example of an Identity Management Realm



In the example in Figure 35-1, the deployment uses a domain name-based DIT structure. The container `dc=us,dc=mycompany,dc=com` is the root of the identity management realm. This results in the creation of a new identity management realm whose scope, by default, is restricted to the entire directory subtree under the entry `dc=us`. The name of the identity management realm is `US`.

35.1.2 Understanding Identity Management Realms in an Enterprise Deployment

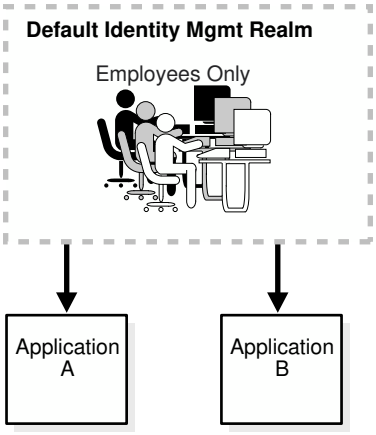
The following topics discuss deployments with single identity management realms and those with multiple ones:

- About Single Identity Management Realm in an Enterprise
- About Multiple Identity Management Realms in an Enterprise

35.1.2.1 About Single Identity Management Realm in an Enterprise

In this scenario, an enterprise has a single set of users, all of whom are managed with the same identity management policies. This is the default configuration of all Oracle products. It includes only one default identity management realm in Oracle Internet Directory and all Oracle components in the enterprise serve users in that realm. Figure 35-2 illustrates this usage.

Figure 35-2 Enterprise Use Case: Single Identity Management Realm

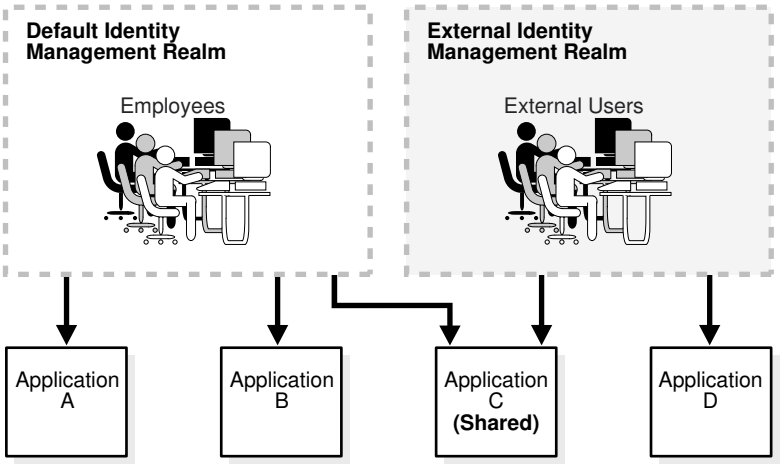


In the example in Figure 35-2, there is a single, default identity management realm containing employees only. In that realm all users and groups are managed and all share access to the same applications, Application A and Application B.

35.1.2.2 About Multiple Identity Management Realms in an Enterprise

You can use the same identity management infrastructure to serve both internal and external self-registered users. Because the identity management policies for internal and external users are different, you can deploy two realms, one for internal and one for external users. Figure 35-3 illustrates this usage.

Figure 35-3 Enterprise Use Case: Multiple Identity Management Realms



In the example in Figure 35-3, the default identity management realm is for internal users—namely, employees—and these have access to Applications A, B, and C. The external identity management realm is for external users, and they have access to Applications C and D.

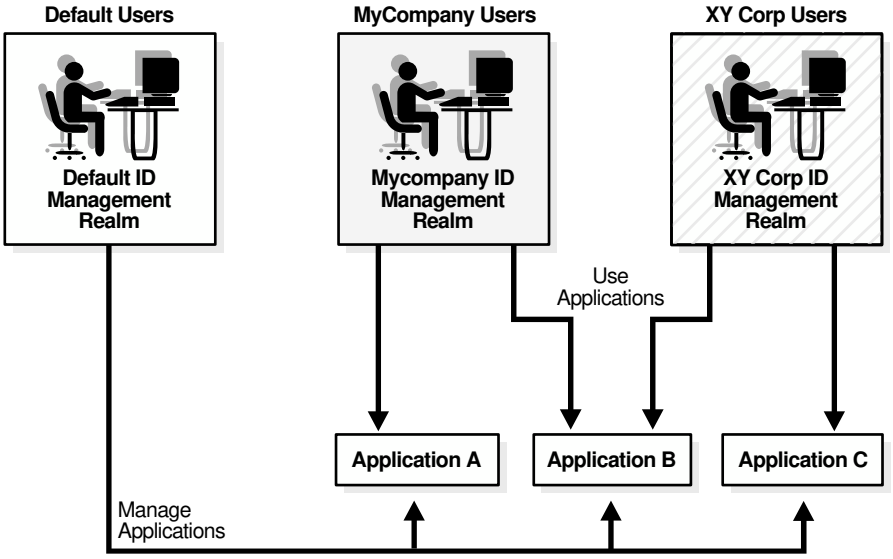
35.1.3 Overview of Identity Management Realms in a Hosted Deployment

In a hosted deployment, the application service provider (ASP) supplies one or more companies with identity management services and hosts applications for them. Each hosted

company is associated with a separate identity management realm where users of that company are managed. Users belonging to the application service provider are managed in a different realm, typically the default realm.

Figure 35-4 shows a hosted deployment with two hosted companies.

Figure 35-4 Hosted Deployment Use Case



In the example in Figure 35-4, the ASP users are in the default identity management realm. The ASP manages its users, groups and associated policies in that realm. ASP users manage Applications A, B, and C for the hosted companies. Hosted company MyCompany users are in the Mycompany identity management realm. They use Applications A and B. Hosted company XY Corp users are in the XY Corp identity management realm. They use Applications B and C.

35.1.4 Identity Management Realm Objects

This section describes the objects in an identity management realm.

Table 35-1 describes the objects in an identity management realm.

Table 35-1 Oracle Identity Management Objects

Object	Description
Root Oracle Context	This object contains: • A pointer to the default identity management realm in the infrastructure • Information on how to locate a realm given a simple name of the realm
Identity Management Realm	A normal directory entry with a special object class associated with it.

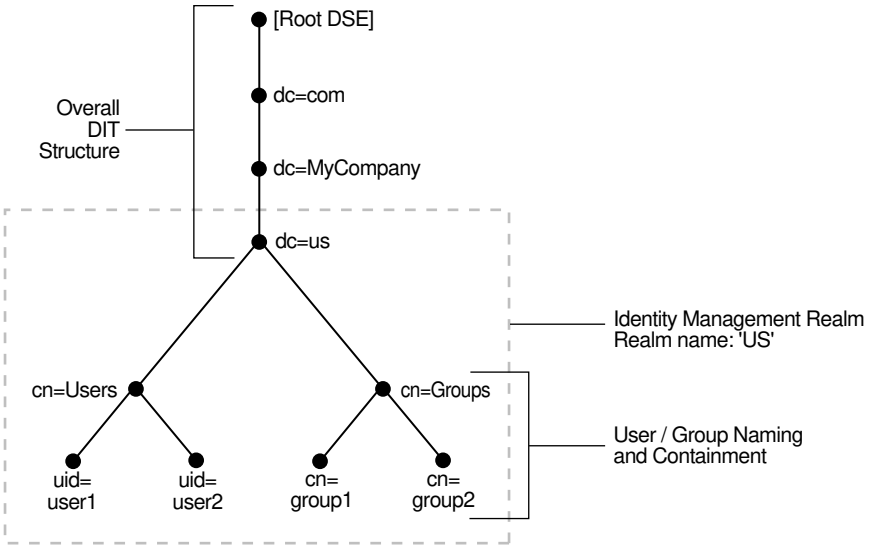
Table 35-1 (Cont.) Oracle Identity Management Objects

Object	Description
Identity Management Realm-Specific Oracle Context	In each realm, this object contains: • User naming policy of the identity management realm—that is, how users are named and located • Mandatory authentication attributes • Location of groups in the identity management realm • Privilege assignments for the identity management realm—for example: who has privileges to add more users to the realm. • Application-specific data for that realm including authorizations

35.1.5 Overview of Default Identity Management Realm

To make configuration easier, Oracle Internet Directory, at installation, creates a default DIT and sets up a default identity management realm.

Figure 35-5 Default Identity Management Realm



As Figure 35-5 shows, the default identity management realm is part of a global DIT. The node just below the root DSE is `dc=com`, followed by `dc=MyCompany`, then `dc=us`. These four nodes represent the overall DIT structure. The node `dc=us` is the root of the default identity management realm. It has two subtrees for containing user and group information: `cn=Users` and `cn=Groups`. For purposes of illustration, the `cn=Users` node contains two leaves: `uid=user1` and `uid=user2`. Similarly, the `cn=Groups` node contains `cn=group1` and `cn=group2`

Oracle Internet Directory gives you the option of setting up a DIT based on the domain of the computer on which the installation is performed. For example, if the installation is on a computer named `oidhost.us.mycompany.com`, then the root of the default identity management realm is `dc=us,dc=mycompany,dc=com`.

It also gives you the option of specifying a different DN that meets your deployment needs as the root of your default identity management realm on the **Specify Namespace in Internet Directory** install screen. For example, if you plan to integrate your Identity Management

installation with a third-party directory, it is recommended that you specify a DN that matches the DN of the default naming context in the third-party directory. For more details on obtaining the default naming context from a third-party directory, refer to the chapter on Connected Directory Integration Concepts and Considerations in *Administering Oracle Directory Integration Platform*.

During configuration, Oracle Internet Directory creates the following:

- An Oracle Context associated with the default identity management realm. The Oracle Context stores all the realm-specific policies and metadata. Using the example in the previous paragraph, it creates the Oracle Context with the distinguished name `cn=OracleContext,dc=us,dc=mycompany,dc=com`. This entry and the nodes under it enable Oracle software to detect realm-specific policies and settings.
- A directory structure and naming policies in the default identity management realm. These enable Oracle components to locate various identities. The default values for these are:
  - All users are located in the container `cn=users` under the base of the identity management realm. In this example, it is `cn=users,dc=us,dc=mycompany,dc=com`.
  - Any new users created in the identity management realm using the Oracle Identity Management infrastructure are also created under the `cn=users` container.
  - All new users created in the identity management realm using the Oracle Identity Management infrastructure belong to the object classes `orclUserV2` and `inetOrgPerson`.
  - All groups are located in the container `cn=groups` under the base of the identity management realm. In this example, it is `cn=groups,dc=us,dc=mycompany,dc=com`.
- Identity management realm administrator. This user is located under the users container. In this example, the fully qualified DN of the realm administrator is `cn=orcladmin,cn=users,dc=us,dc=mycompany,dc=com`.

> **Note:**
>
> If the realm administrator's account becomes locked, the Oracle Internet Directory superuser can unlock it by modifying the realm administrator's account password, using Oracle Directory Manager.

- Default authentication policies, which enable authentication services to perform the appropriate actions. These include:
  - The default directory password policy—for example, password length, lockout, and expiration
  - Additional password verifiers that need to be automatically generated when provisioning the user
- Identity management authorizations. Oracle Internet Directory grants these to the realm administrator who can further delegate these authorizations through the Oracle Internet Directory Self-Service Console. Some of these authorizations include:
  - Common identity management configuration privileges—for example, user creation, user profile modification, group creation
  - Privileges to install new Oracle components by using the Oracle Identity Management infrastructure.
  - Privileges to administer Oracle Internet Directory Self-Service Console

See Also:

* `orclUserV2` object class in *Reference for Oracle Identity Management* for more information about the `orclUserV2`
* Delegating Privileges for Oracle Identity Management for a fuller description of the default access control policies in Oracle Identity Management

35.2 Customizing the Default Identity Management Realm

The following topics describe how to customize the default identity management realm based on a number of use cases:

- Default Identity Management Realm Customization
- Understanding the Use Cases for Default Identity Management Realm Customization
- Updating the Existing User and Group Search Base
- Setting Up an Additional Search Base
- Refreshing the Oracle Single Sign-On
- Reconfiguring the Provisioning Profiles

35.2.1 Default Identity Management Realm Customization

After a realm is created, you can further customize various aspects of it.

Table 35-2 lists the aspects you can customize, the tools available for each type of customization, and where to look for more information.

Table 35-2 Customizing the Default Identity Management Realm

What You Can Customize	Tools	Information
Directory structure and naming policies	Oracle Internet Directory Self-Service Console Oracle Directory Services Manager Command-line tools	This section. Overview of Default Identity Management Realm The chapter on using the *Oracle Internet Directory Self-Service Console* in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager*.
Authentication policies	Oracle Directory Services Manager Command-line tools	Managing Password Policies
Identity management authorizations	Oracle Internet Directory Self-Service Console Oracle Directory Services Manager Command-line tools	Delegating Privileges for Oracle Identity Management The chapter on using the Oracle Internet Directory Self-Service Console in *Oracle Identity Management Guide to Delegated Administration*.

35.2.2 Understanding the Use Cases for Default Identity Management Realm Customization

A typical scenario where this might be required is one where you must integrate your Oracle Identity Management installation with a third-party directory.

For example, assume the default Identity Management Realm is `dc=mycompany,dc=com` and there are users under `cn=users,dc=mycompany,dc=com`.

If the third party directory naming context does not match the current user and group search base in the default realm, then you can alter the user and group search base of the default realm so that both the existing users and the third party users can log in using Oracle Single Sign-On. Select a user search base just high enough to include the existing users and the third party users. Let us call this search base the Lowest Common User Search Base.

> **Note:**
>
> This approach assumes that the user `nickname` attribute selected for Oracle Single Sign-On Login is unique across the existing user search base and the third party directory naming context. Otherwise, Oracle Single Sign-On authentication fails for all those users whose `nickname` attribute values clash.

If your deployment scenario matches any of the use cases from 1 to 5, follow the procedure described in Updating the Existing User and Group Search Base.

Use Case 1

The third party naming context is under the default realm, but in a different container than the realm user search base

For example, the existing users are under `cn=users,dc=mycompany,dc=com` and the third party naming context is under `cn=users,o=employees,dc=mycompany,dc=com`. In this case, the Lowest Common User Search Base is `dc=mycompany,dc=com`.

Use Case 2

The third party naming context is outside the default realm, but there is a Lowest Common User Search Base.

For example, the existing users are under `cn=users,dc=mycompany,dc=com` and the third party naming context is under `cn=users, dc=mycompanyecorp,dc=com`. In this case, the Lowest Common User Search Base is `dc=com`.

Use Case 3

The third party naming context is the same as the default realm DN.

For example, the existing users are under `cn=users,dc=mycompany,dc=com` and the third party naming context is directly under `dc=mycompany,dc=com`. In this case, the Lowest Common User Search Base is `dc=mycompany,dc=com`.

Use Case 4

The third party naming context contains the parent of the default realm DN.

For example, you might have a default realm with `DN: dc=us,dc=mycompany,dc=com`, existing users under `cn=users,dc=us,dc=mycompany,dc=com` and the third party naming context directly under `dc=com`. In this case, the Lowest Common User Search Base is `dc=com`.

Use Case 5

The third party naming context is under the existing user search base.

For example, existing users are under `cn=users,dc=mycompany,dc=com` and the third party naming context is directly under `l=emea,cn=users,dc=mycompany,dc=com`. In this case, the Lowest Common User Search Base is `cn=users,dc=mycompany,dc=com`. In this use case, you do not need to change the user search base.

Use Case 6

In this case, the third party naming context is outside the default realm and the Lowest Common User Search Base is the root DSE.

For example, if existing users are under `cn=users,dc=mycompany,dc=com` and the third party naming context is under `cn=users,dc=mycompanycorp,dc=net`, then the Lowest Common User Search Base is the root DSE.

In this case, you must add the third party naming context as an additional search base. The steps are as follows:

1. "Setting Up an Additional Search Base"
2. "Refreshing the Oracle Single Sign-On"
3. "Reconfiguring the Provisioning Profiles"

35.2.3 Updating the Existing User and Group Search Base

This section describes the procedure to update the existing user group search base.

To set up synchronization with the third party directory:

1. Back up the Oracle Internet Directory database.
2. Create the user and group containers for the third party directory, using Oracle Directory Services Manager, if the entries do not already exist in the directory.
3. Apply appropriate ACLs on the new users container by doing the following:
   a. Instantiate the variables `%USERBASE%` and `%REALMBASE%` in the ACL template file `$ORACLE_HOME/ldap/schema/oid/oidUserAdminACL.sbs` and create the file `usracl.ldif`. Set the variable `%USERBASE%` to the DN of the new user container and the variable `%REALMBASE%` to the default realm DN.
   b. Upload the instantiated LDIF file `usracl.ldif` using the `ldapmodify` command.
4. Apply appropriate ACLs on the new groups container by doing the following:
   a. Instantiate the variables `%GRPBASE%` and `%REALMBASE%` in the ACL template file `$ORACLE_HOME/ldap/schema/oid/oidGroupAdminACL.sbs` and create the file `grpacl.ldif`. Set the variable `%USERBASE%` to the DN of the new user container and the variable `%REALMBASE%` to the default realm DN.
   b. Upload the instantiated LDIF file `grpacl.ldif` using the `ldapmodify` command.
5. Determine a Lowest Common User Search Base base that is just high enough to include the existing users and the third party users.

For example, if existing users are under `cn=users,dc=mycompany,dc=com` and the third party users are under `l=emea,dc=mycompany,dc=com`, then the Lowest Common User Search Base is `dc=mycompany,dc=com`.

The Lowest Common User Search Base might be the root entry. That is the case if, for example, the existing users are under `cn=users,dc=mycompany,dc=com` and the third party users are under `dc=mycompanycorp,dc=net`. In that case, skip to the deployment scenario described in Setting Up an Additional Search Base.

6. If you must also synchronize groups, determine a group search base that is just high enough to include the existing groups and the third party groups. Lets call this search base the Lowest Common Group Search Base.

   For example, if existing groups are under `cn=groups,dc=mycompany,dc=com` and the third party groups are under `l=emea,dc=mycompany,dc=com`, then the Lowest Common Group Search Base is `dc=mycompany,dc=com`.

7. Log in to the Self-Service Console as the administrator of the realm (usually `orcladmin`).
8. Go to the **Configuration** tab and set the user search base to the Lowest Common User Search Base you determined. If you must also synchronize groups, then also then set the group search base to the Lowest Common Group Search Base that you determined.
9. To make Oracle Single Sign-On recognize these changes, follow the procedure described under Refreshing the Oracle Single Sign-On.
10. Verify the Oracle Single Sign-On login of users in the original user search base by logging in as `orcladmin`.
11. You must also reconfigure the applications that have been provisioned to reflect the modified user and group bases. Follow the steps described under Reconfiguring the Provisioning Profiles.

> **Note:**
>
> In addition to the user and group search base attributes, you can also modify other configuration settings of an identity management realm, such as the attribute for Login Name (nickname) or the attribute for RDN, using the Self-Service Console. See: "Modifying Configuration Settings for an Identity Management Realm" in the chapter "Using the Oracle Internet Directory Self-Service Console" of *Oracle Identity Management Guide to Delegated Administration* in the 10g (10.1.4.0.1) library for more details.

35.2.4 Setting Up an Additional Search Base

This section describes the procedure to set up an additional search base.

To set up synchronization with the third party directory:

1. Back up the Oracle Internet Directory database.
2. Create the user and group containers for the third party directory, using Oracle Directory Services Manager, if the entries do not already exist in the directory.
3. Apply appropriate ACLs on the new users container by doing the following:
   a. Instantiate the variables `%USERBASE%` and `%REALMBASE%` in the ACL template file `$ORACLE_HOME/ldap/schema/oid/oidUserAdminACL.sbs` and create the file

`usracl.ldif`. Set the variable `%USERBASE%` to the DN of the new user container and the variable *%REALMBASE%* to the default realm DN.

b. Upload the instantiated LDIF file `usracl.ldif` using the `ldapmodify` command.

4. Apply appropriate ACLs on the new groups container by doing the following:

a. Instantiate the variables `%GRPBASE%` and `%REALMBASE%` in the ACL template file `$ORACLE_HOME/ldap/schema/oid/oidGroupAdminACL.sbs` and create the file `grpacl.ldif`. Set the variable `%USERBASE%` to the DN of the new user container and the variable `%REALMBASE%` to the default realm DN.

b. Upload the instantiated LDIF file `grpacl.ldif` using the `ldapmodify` command.

5. Log in to the Self-Service Console as the administrator of the realm.

6. Go the **Configuration** tab.

a. Add `cn=users,dc=mycompanycorp,dc=net` to the `usersearchbase` for the current realm.

b. Add `cn=groups,dc=mycompanycorp,dc=net` to the `groupsearchbase` for the current realm.

7. To make Oracle Single Sign-On recognize these changes, follow the procedure described under Refreshing the Oracle Single Sign-On.

8. Verify the Oracle Single Sign-On login of users in the original user search base by logging in as `orcladmin`.

9. If mid-tiers have been configured against this identity management configuration, then you must also reconfigure the applications that have been provisioned to reflect the modified user and group bases. Follow the steps described under Reconfiguring the Provisioning Profiles.

> **Note:**
>
> In addition to the user and group search base attributes, you can also modify other configuration settings of an identity management realm, such as the attribute for Login Name (nickname) or the attribute for RDN, using the Self-Service Console. See: "Modifying Configuration Settings for an Identity Management Realm" in the chapter "Using the Oracle Internet Directory Self-Service Console" of *Oracle Identity Management Guide to Delegated Administration* in the 10g (10.1.4.0.1) library for more details.

35.2.5 Refreshing the Oracle Single Sign-On

This section describes how to refresh Oracle Single Sign-on script.

To make Oracle Single Sign-On recognize your configuration changes, execute the Oracle Single Sign-On refresh script by changing to the directory `$ORACLE_HOME/sso/admin/plsql/sso/` and typing:

```
sqlplus orasso@ssoreoid.sql
```

you are prompted for the password. To get the `orasso` schema password, refer to the appendix "Obtaining the Single Sign-On Schema Password" *in the Oracle Application Server Single Sign-On Administrator's Guide* in the 10g (10.1.4.0.1) library.

35.2.6 Reconfiguring the Provisioning Profiles

If you installed middle-tier applications against this default identity management realm before changing its user and group search bases, then the provisioning profiles created by the middle-tier installations become invalid. This happens because the profiles have the old user or group search base information in the `event subscriptions` attribute. You must modify all the profiles by using `oidprovtool`.

To reconfigure every provisioning profile:

1. Use `ldapsearch` to put all the provisioning profile information into an LDIF file:

```
ldapsearch -h oid_host -p oid_port \
           -D "cn=orcladmin" -q -s sub \
           -b "cn=provisioning profiles,cn=changelog subscriber,\
               cn=oracle internet directory" \
           "objectclass=*" >  provprofiles.ldif
```

   The event subscriptions look something like this:

```
USER:cn=users,dc=mycompany,dc=com:MODIFY(list_of_attributes)USER:cn=users,dc=mycompan
y,dc=com:DELETEGROUP:cn=groups,dc=mycompany,dc=com:MODIFY(list_of_attributes)GROUP:cn
=groups,dc=mycompany,dc=com:DELETE
```

   where `cn=users,dc=mycompany,dc=com` and `cn=groups,dc=mycompany,dc=com` are the user and group search bases, respectively, that were created when you installed and configured the application.

2. Get the actual DNs of the application identity by searching the Oracle Internet Directory server based on the GUID. To get the application DN, type:

```
ldapsearch -h host -p port -D cn=orcladmin -q \
           -s sub -b "" \
           "orclguid=Value_of_orclODIPProvisioningAppGuid" dn
```

   You can get the GUID values for each profile from the attribute values in `provprofiles.ldif`.

3. Modify each of the returned profiles as follows:

```
$ORACLE_HOME/bin/oidprovtool operation=MODIFY \
        ldap_host=host ldap_port=port \
ldap_user_dn="cn=orcladmin" \
ldap_user_passwd=password \
interface_version=interfaceVersion \
        application_dn=applicationDN \
        organization_dn=identity_Realm_DN \
        event_subscription=New_Event_Subscription_1
        event_ subscription=New_Event_Subscription_2
        .
        .
        .
        event_subscription=New_Event_Subscription_n
```

   The `New_Event_Subscription` arguments should be of the form:

```
USER: new_user_search_base:MODIFY(list_of_attributes)
USER: new_user_search_base:DELETE
GROUP: new_group_search_base:MODIFY(list_of_attributes)
GROUP: new_group_search_base:DELETE
```

Here, the `organization_dn` value should be the original identity realm DN

35.3 About Additional Identity Management Realms for Hosted Deployments

You can create additional identity management realms by using the Self-Service Console in Oracle Identity Management by using `oidrealm`.

Only members of the ASPAdmins group can create a new identity management realm. Use Oracle Directory Services Manager to add a user to that group by adding the userDN to the uniquemember attribute of group ASPAdmins in the Default Identity Management Realm-specific OracleContext. Refer to the section on "Modifying a Static Group Entry by Using Oracle Directory Services Manager" for details.

Note:

Not all applications can work with multiple identity management realms.

Whenever you add an additional realm, you may need to make existing applications aware of it by using a manual procedure. For more information, see the application-specific documentation.

In the Oracle Identity Management infrastructure, the single sign-on server must be made aware of an additional realm by using a special administrative procedure. Please refer to the chapter "Single Sign-On in Multiple Realms" in the *Oracle Application Server Single Sign-On Administrator's Guide* for instructions on enabling multiple realms in Oracle Single Sign-On.

See Also:

- The `oidrealm` command reference in *Reference for Oracle Identity Management*.
- Modifying an Attribute of a Static Group Entry.
- See chapter on the Oracle Internet Directory Self-Service Console in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager* for information about creating an additional identity management realm.

35.4 Creating a DIT View

Beginning with 11g Release 1 (11.1.1.9.0), you can create a DIT view, which is a virtual view or name space that shows entries from a different or source DIT. The following items are transformed from the source DIT to the DIT view name space:

- DN and all DN attributes in the entry
- RDN mappings that are specified for the DIT view

A DIT view reduces the overall storage requirements in a cloud environment for common entries across tenants. All operations except LDAPMODDN are allowed for a DIT view. A DIT

view also allows you to derive the RDN attribute value from any attribute in the entry, so you can hide the original entry when user names are visible in the entry DN.

The section includes the following information:

- Specifying a DIT View
- About DIT View Conditions
- Examples of DIT View

35.4.1 Specifying a DIT View

You can specify the DIT view using the following command:

To specify a DIT view, use the following syntax:

```
objectclass: orclditview
objectclass: extensibleobject
orclRDNAttrMapping:targetAttribute:sourceAttribute:valueFromAttribute[,optionalRelativeDN
]
orclDITBaseDN: realDN
orclrdnmapping: targetRDN:sourceRDN
orclreturnattr: orclmemberof
orclreturnattr: attrName
```

35.4.2 DIT View Syntax

This section describes items in the `orclditview` object class.

Table 35-3 describes the items used to specify a DIT view.

Table 35-3 DIT View Syntax

Item	Description
`objectclass: orclditview`	Indicates to Oracle Internet Directory server to process the entry for DIT view mapping.
`objectclass: extensibleobject`	Allows the entry to optionally hold any attribute.
`orclRDNAttrMapping: targetAttribute:sourceAttribute:valueFromAttribute[,optionalRelativeDN ]`	Specifies the mapping of a source attribute to a target attribute and the option of using a value from a different attribute: • *`targetAttribute`* is the name of the RDN attribute of the entry in the DIT view. • *`sourceAttribute`* is the name of the RDN attribute in the original source entry. • *`valueFromAttribute`* is the attribute from which the RDN value is derived. In most cases, this value can be the same as *`sourceAttribute`*. • *`optionalRelativeDN`* optionally specifies if the server performs the transformation only if under this relative DN.
`orclDITBaseDN: realDN`	Specifies the *`realDN`*, which is the location of the original source entries. For example: `ou=people,dc=us,dc=example,dc=com`

Table 35-3 (Cont.) DIT View Syntax

Item	Description
`orclrdnmapping: targetRDN: sourceRDN`	Specifies RDN mapping from the *sourceRDN* to the destination *targetRDN*.
`orclreturnattr: attrName`	Specifies multi-valued attribute names to return. Oracle Internet Directory server returns only these configured attributes. Optional. If *orclreturnattr* is not specified, the server returns all attributes of the entry.
`orclmaskfilter: LDAP Filter`	Use this LDAP filter to mask entries.
`orclrdnfilter: LDAP Filter`	Use this LDAP filter to perform RDN mapping conversion only if `orclrdnfilter` is applicable.
`orclattrmapping:targetAttr:SourceAttr:overwrite`	Copies value from `SourceAttr` to `target Attr`. If you set `overwrite` to `1`, then `targetAttr` values only contain values from `sourceAttr`. Otherwise it is appended to existing `targetAttr` values.
`orclattrexclude: attrName: specificValue`	Allows you to exclude an attribute or a specific value of the attribute. The `specificValue` parameter is optional.
`orclalternateDITDN: AlternateDN`	Allows you to search for entries under different container when it is not found in `orclDITBaseDN`. It is single-valued. configuration. Note: You must configure `orclmaskfilter` to allow Oracle Internet Directory server to distinguish when to use `orclDITBaseDN` versus `orclalternateDITDN`.

35.4.3 About DIT View Conditions

This section introduces you to DIT view conditions.

You must keep the following conditions in mind while working with DIT views:

- The RDN attribute of the entry in DIT view that gets mapped must not contain special characters, such as comma (,) , plus (+), equal (=), semicolon (;), hash (#), backslash (\), less than (<), and greater than (>).
- The RDN attribute of the entry in DIT view that is mapped must be a single-valued attribute with `uniqueness` attribute enabled.
- If you wish to obtain the aliases (`memberof`/`ismemberof`) of the `orclmemberof` attribute from the DIT view, then while creating the DIT view you must specify the alias in the return attribute list as follows:

  ```
  orclreturnattr: orclmemberof
  ```

35.4.4 Examples of DIT View

In the following example, the real entry will be transformed to the DIT view entry.

Real entry:

```
uid=user.1,ou=people,dc=us,dc=example,dc=com
```

DIT view entry:

```
displayname=user.1,ou=people,dc=uk,dc=example,dc=com
```

For example:

```
dn: ou=people,dc=uk,dc=example,dc=com
objectclass: top
objectclass: orclditview
objectclass: extensibleobject
orclrdnattrmapping: displayname:uid:mail
orclditbasedn: ou=people,dc=us,dc=example,dc=com
```

In the following example, the real entry will be transformed to the DIT mapped entry, where `user.1@example.com` is the `mail` attribute value.

Real entry:

```
uid=user.1,ou=people,dc=us,dc=example,dc=com
```

DIT view entry:

```
displayname=user.1@example.com,ou=people,dc=ind,dc=example,dc=com
```

For example:

```
dn: ou=people,dc=ind,dc=example,dc=com
objectclass: top
objectclass: orclditview
objectclass: extensibleobject
orclrdnattrmapping: displayname:uid:mail
orclditbasedn: ou=People,dc=us,dc=example,dc=com
ou: people
```

In the following example, the real entry will be transformed to the DIT mapped entry, where `user.1@example.com` is the mail attribute value.

Real entry:

```
uid=user.1,ou=people,dc=us,dc=example,dc=com
```

DIT view entry:

```
displayname=user.1@example.com,cn=users,cn=common,dc=example,dc=com
```

For example:

```
dn: cn=common,dc=example,dc=com
objectclass: top
objectclass: orclditview
objectclass: extensibleobject
orclrdnattrmapping: displayname:uid:mail
orclditbasedn: dc=us,dc=example,dc=com
orclrdnmapping: cn=users:ou=people
orclrdnmapping: cn=groupview:cn=groups
cn: common
```

The following example returns an attribute configuration. The configuration shows only the `uid`, `mail`, `cn`, and `sn` attributes to be returned when entries are searched under this tree. Note that while adding or updating the entry with any other attributes, this restriction does not apply, which allows entries with other attributes to be added.

```
dn: cn=common,dc=example,dc=com
objectclass: top
objectclass: orclditview
objectclass: extensibleobject
orclrdnattrmapping: displayname:uid:mail
orclditbasedn: dc=us,dc=example,dc=com
orclrdnmapping: cn=users:ou=people
orclrdnmapping: cn=groupview:cn=groups
orclreturnattr: cn
orclreturnattr: uid
orclreturnattr: mail
orclreturnattr: sn
cn: common
```

The following example describes attribute mapping. The syntax is as follows:

```
orclattrmapping=targetAttr:SourceAttr:overWrite
```

Now, if `overWrite` is set as `1` then existing values of `targetAttr` are removed. If `overWrite` is set as `0`, then `sourceAttr` values are appended to existing values of `targetAttr`.

```
dn: cn=ditview,dc=oracle,dc=com
changetype: modify
replace: orclattrmapping
orclattrmapping: cn:mail:0
orclattrmapping: sn:givenname:1
```

The following example describes how to exclude a value of an attribute. The syntax is as follows:

```
orclattrexclude: attribute: optionalValue

dn: cn=ditview,dc=oracle,dc=com
changetype: modify
orclattrexclude: objectclass:orgperson
orclattrexclude: authpassword
orclattrexclude: userpassword
```

The following example describes alternate base DN.

```
dn: cn=filteredusers,dc=oracle
objectclass: top
objectclass: orclditview
objectclass: extensibleobject
orclditbasedn: ou=people,dc=us,dc=oracle,dc=com
orclalternateditdn: ou=contractors, dc=us,dc=oracle,dc=com
orclmaskfilter: (employeetype=regular)
cn: filteredusers
```

36
Tuning and Sizing Oracle Internet Directory

Recommendations for sizing and tuning Oracle Internet Directory are documented in the Performance Planning chapter in *Tuning Performance*.

> **Note:**
>
> Oracle Internet Directory's out of box configuration is not optimal for most production or test deployments. You must follow at least the basic tuning steps listed in the Oracle Internet Directory Performance Tuning chapter in *Tuning Performance* to achieve optimal performance and availability.

37
Managing Garbage Collection

The following topics describe how to manage garbage collection in Oracle Internet Directory, including setting the Oracle Database time zone for garbage collection, managing the Oracle Internet Directory garbage collectors, and managing the logging for the garbage collectors:

- Understanding Garbage Collection Management
- Setting Oracle Database Time Zone for Garbage Collection
- Modifying the Oracle Internet Directory Garbage Collectors
- Managing Oracle Internet Directory Garbage Collectors Logging
- Configuring Time-Based Change Log Purging

The term "garbage" refers to any data not needed by the directory but still occupying space in it. This unwanted or obsolete data can eventually fill up the disk and decrease directory performance. The process of removing this unwanted data from the directory is called garbage collection.

37.1 Understanding Garbage Collection Management

A garbage collector is a background database process that removes unwanted data from the directory.

The Oracle Internet Directory garbage collection framework provides a default set of garbage collectors, and enables you to modify them. The Oracle Internet Directory statistics collector also uses the Oracle Internet Directory garbage collection framework.

This introduction contains these topics:

- Understanding the Components of the Garbage Collection Framework
- How Oracle Internet Directory Garbage Collection Works
- About Garbage Collector Entries and Statistics Collector Entry
- Overview of Change Log Purging

37.1.1 Understanding the Components of the Garbage Collection Framework

The following topics describe the components that make up the Oracle Internet Directory garbage collection framework, namely, the garbage collection plug-in and the background database processes.

- About Garbage Collection Plug-in
- Understanding the Background Database Processes

37.1.1.1 About Garbage Collection Plug-in

Garbage collection in Oracle Internet Directory relies on a garbage collection plug-in that receives requests to manage garbage collectors. This plug-in is installed with Oracle Internet

Directory, and is enabled by default. The entry for this plug-in is `cn=plugin,cn=subconfigsubentry`.

This plug-in has three triggers:

- The DN of the plug-in trigger used to create a garbage collection job is: `cn=Add PurgeConfig,cn=plugin,cn=subconfigsubentry`.
- The DN of the plug-in trigger used to modify a garbage collection job is: `cn=Modify PurgeConfig,cn=plugin,cn=subconfigsubentry`.
- The DN of the plug-in trigger used to delete a garbage collection job is: `cn=Delete PurgeConfig,cn=plugin,cn=subconfigsubentry`.

> **See Also:**
>
> Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management* for a list and descriptions of the attributes of the garbage collection plug-in

37.1.1.2 Understanding the Background Database Processes

The following topics describe the background database processes that are invoked by the garbage collection plug-in, which include the garbage collectors and the Oracle Internet Directory statistics collector:

- About Garbage Collectors
- About Predefined Garbage Collectors
- About Statistics Collector

37.1.1.2.1 About Garbage Collectors

You can set and manage these behaviors of a garbage collector:

- The time it starts
- The age of the data you want it to purge
- How often it runs
- The type of data you want it to purge
- The number of entries to purge at a time

37.1.1.2.2 About Predefined Garbage Collectors

A default installation of Oracle Internet Directory includes these predefined garbage collectors:

- Change log garbage collector: Cleans up the consumed change log entries in the directory. The container for this garbage collector is `cn=changelog purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- General statistics garbage collector: Cleans up obsolete entries created by Oracle Internet Directory Server Manageability for monitoring general statistics of the directory. The container for this garbage collector is `cn=general stats purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.

- Health statistics garbage collector: Cleans up obsolete entries created by Oracle Internet Directory Server Manageability for monitoring health statistics of the directory. The container for this garbage collector is `cn=health stats purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- Security and refresh events garbage collector: Cleans up obsolete entries created by Oracle Internet Directory Server Manageability for monitoring security and refresh events of the directory. The container for this garbage collector is `cn=secrefresh events purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- System resource events garbage collector: Cleans up obsolete entries created by Oracle Internet Directory Server Manageability for monitoring system resource events of the directory. The container for this garbage collector is `cn=sysresource events purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- Tombstone garbage collector: Cleans up obsolete entries marked as deleted in the directory. The container for this garbage collector is `cn=tombstone purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- LDAP performance monitoring garbage collector: Cleans up LDAP server performance statistics data. The container for this garbage collector is `cn=perf stats purgeconfig,cn=purgeconfig,cn=subconfigsubentry`.
- LDAP bind performance monitoring garbage collector: Cleans up bind performance data gathered for security events tracking. The container for this garbage collector is `cn=bindsec stats purgeconfig,cn=purgeconfig, cn=subconfigsubentry`.
- LDAP compare performance monitoring garbage collector: Cleans up compare performance data gathered for security events tracking. The container for this garbage collector is `cn=comparesec stats purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.
- LDAP compare failure performance monitoring garbage collector: Cleans up compare failure performance data gathered for security events tracking. The container for this garbage collector is `cn=comparefailure stats purgeconfig,cn=purgeconfig,cn=subconfigsubentry`.

See Also:

- Capabilities of Oracle Internet Directory Server Manageability
- Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management* .

Note:

Oracle recommends that you not delete any of the predefined garbage collectors. Deleting one or more of them can result in the proliferation of obsolete data, eventually exhausting all the available disk space.

You may, however, modify predefined garbage collectors to customize their behavior.

37.1.1.2.3 About Statistics Collector

You can set and manage these behaviors of the Oracle Internet Directory statistics collector:

- The time it starts
- How often it runs

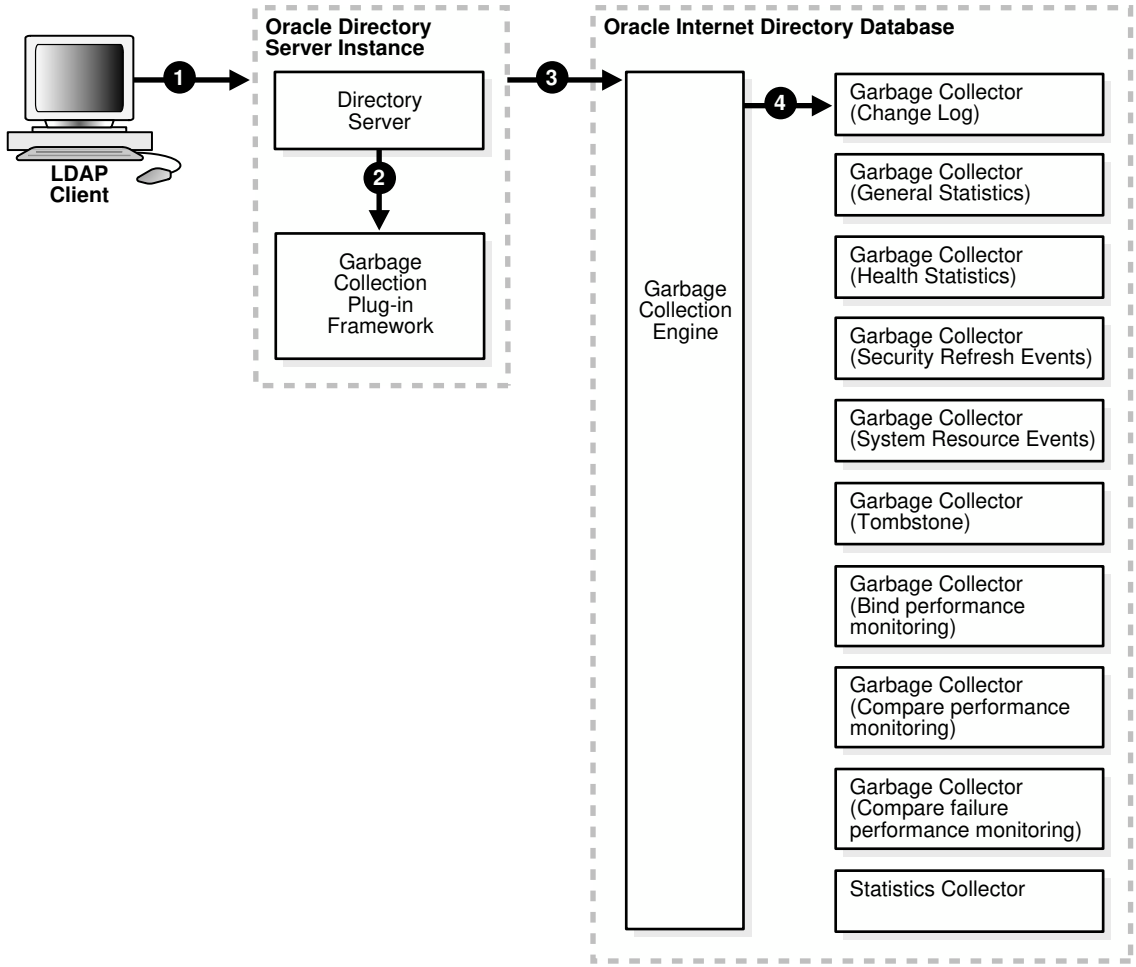
The Oracle Internet Directory statistics collector collects statistics about Oracle Internet Directory. The container for this background database process is `cn=oidstats_config, cn=purgeconfig,cn=subconfigsubentry`.

37.1.2 How Oracle Internet Directory Garbage Collection Works

This section depicts an example of a garbage collector operation that purges change log entries.

Figure 37-1 shows an example of a garbage collector operation that purges change log entries.

Figure 37-1 Example: Garbage Collection of Change Log Entries



As the example in Figure 37-1 shows, the garbage collection process is as follows:

1. An LDAP client sends to the directory server a request for a particular garbage collection operation. The operation could be, for example, to purge the entries of tombstone or, change logs.

2. The directory server passes the request to the garbage collection plug-in.
3. The garbage collection plug-in sends the request to the garbage collection engine in the Oracle Internet Directory-designated database.
4. The garbage collection engine triggers the corresponding background database process—in this case, the change log garbage collector. The background database process runs according to the parameters specified in its configuration.

37.1.3 About Garbage Collector Entries and Statistics Collector Entry

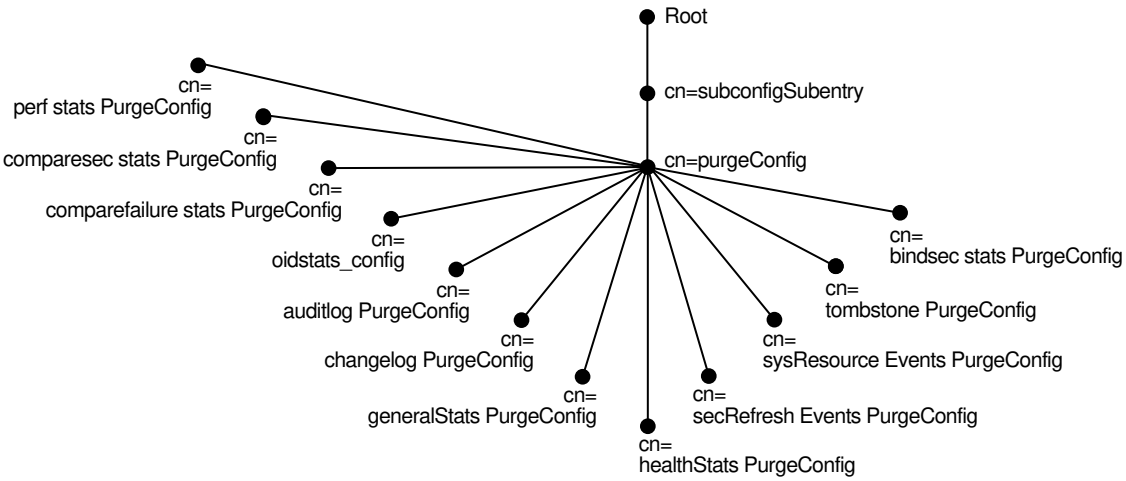
Garbage collector entries, each with attributes specifying how it is to behave, are located in the entry `cn=purgeconfig`, which is located immediately below the entry `cn=subconfigsubentry`.

> **See Also:**
>
> Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management* for a description of each garbage collector attribute

The Oracle Internet Directory statistics collector entry, with its attributes, is also located in the entry `cn=purgeconfig`, immediately below the entry `cn=subconfigsubentry`.

Figure 37-2 shows the location of these entries.

Figure 37-2 Garbage Collection Entries in the DIT



37.1.4 Overview of Change Log Purging

Both replication and Oracle Directory Integration Platform use change logs to propagate information from a supplier directory to a consumer directory.

All change logs are stored in the table `ods_chg_log`. In addition, replication change logs are stored in `asr_chg_log`. When the change log garbage collector runs, it purges change logs that are no longer needed by any change log consumers. This prevents the change log store in the Oracle Internet Directory database from becoming too large.

The change log garbage collector uses the following two methods to determine which change logs to purge:

- Change number-based purging

  Change number-based purging respects the change status of all change log consumers. That is, it does not purge change logs unless they have been consumed by all consumers. When the change log garbage collector runs, it purges all change logs that have been consumed by replication, Oracle Directory Integration Platform, and other consumers.

- Time-based purging

  Time-based purging is a fall-back method designed to purge change logs of a certain age. It ensures that old change logs are purged even if they have not been consumed by all change log subscribers. Time-based purging respects the change status of replication, but not the change status of other consumers. The change log garbage collector purges all change logs that are not needed by replication and that are at least `orclpurgetargetage` hours old. If `orclpurgetargetage` is zero, the change log garbage collector does this immediately. If `orclpurgetargetage` is an invalid number or not defined, the default value is 240 hours (10 days). Change logs needed by replication are not purged until they have been consumed by replication.

  If you have deployed Oracle Directory Integration Platform, and you want to enable time-based purging, be sure to set `orclpurgetargetage` to a large enough value to allow change logs to be processed by Oracle Directory Integration Platform before they are purged. A value of 240 allows 10 days before change logs are purged.

  See Configuring Time-Based Change Log Purging.

37.2 Setting Oracle Database Time Zone for Garbage Collection

To ensure that the Oracle Internet Directory garbage collection logic works correctly, you must set the Oracle Database `dbtimezone` parameter to the appropriate displacement from Coordinated Universal Time (UTC).

Perform the following steps:

1. Invoke PL/SQL:

   ```
   $ sqlplus /nolog
   SQL> connect / as sysdba
   ```

2. Perform a query to get the value of `dbtimezone`:

   ```
   SQL> select dbtimezone from dual;
   ```

3. Perform a query to get the displacement from Coordinated Universal Time (UTC):

   ```
   SQL> select systimestamp from dual;
   ```

   The output will look similar to this:

   ```
   SYSTIMESTAMP
   ---------------------------------------
   31-JAN-09 01.04.42.281000 PM -05:00
   ```

   Get the last column of the `systimestamp` output. In the example, it is `-05:00`.

4. If the `dbtimezone` parameter is equal to last column value of the `systimestamp` output, you do not need to perform the remaining steps. Otherwise, proceed.

5. Stop all instances of Oracle Internet Directory that are using the Oracle Database, as described in Managing Oracle Internet Directory Instances.
6. Set the `dbtimezone` parameter, using the value you got from the last column the `systimestamp` query:

   ```
   SQL> ALTER DATABASE SET TIME_ZONE = '-05:00';
   ```

7. Shut down the Oracle Database:

   ```
   SQL> shutdown immediate
   ```

8. Restart the Oracle Database:

   ```
   SQL> startup
   ```

9. Restart Oracle Internet Directory as described in Managing Oracle Internet Directory Instances.

> **Note:**
>
> *Oracle OLAP DML Reference* for more information about the `dbtimezone` parameter.

37.3 Modifying the Oracle Internet Directory Garbage Collectors

The following topics describe how to modify the Oracle Internet Directory Garbage collector and the Oracle Internet Directory Statistics Collector:

This section has the following topics:

- Modifying a Garbage Collector by Using Oracle Directory Services Manager
- Modifying a Garbage Collector by Using Command-Line Tools
- Modifying the Oracle Internet Directory Statistics Collector

37.3.1 Modifying a Garbage Collector by Using Oracle Directory Services Manager

Using Oracle Directory Services Manager, you can modify a garbage collector.

To modify a garbage collector:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Advanced**.
3. Expand **Garbage Collection** in the left pane, then select the garbage collector you want to modify. The Garbage Collector Window appears in the right pane.
4. In the **Garbage Collector** window, enter the changes you want to make for this garbage collector.
5. Choose **Apply**.

37.3.2 Modifying a Garbage Collector by Using Command-Line Tools

Using examples understand how to modify garbage collectors by using command-line tools.

This section contains the following topics:

- Modifying a Garbage Collector
- Disabling a Garbage Collector Change Log

The garbage collection attributes that you can modify are listed in Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management*.

37.3.2.1 Modifying a Garbage Collector

Suppose that you want the tombstone garbage collector to run immediately. The LDIF would look like this:

```
dn: cn=tombstone purgeconfig, cn=purgeconfig, cn=subconfigsubentry
changetype:modify
replace: orclpurgenow
orclpurgenow: 1
```

Load this entry with ldapmodify.

```
ldapmodify -D "cn=orcladmin" -q -h hostname -p port \
   -D username -f file_name_of_defined_entry
```

37.3.2.2 Disabling a Garbage Collector Change Log

Suppose that you want to disable changelog garbage collector.

```
dn: cn=changelog purgeconfig, cn=purgeconfig, cn=subconfigsubentry
changetype: modify
replace: orclpurgeenable
orclpurgeenable: 0
```

Load this entry with ldapmodify.

```
ldapmodify -D "cn=orcladmin" -q -h hostname -p port \
   -D username -f file_name_of_defined_entry
```

37.3.3 Modifying the Oracle Internet Directory Statistics Collector

You modify the Oracle Internet Directory statistics collector.

You modify the Oracle Internet Directory statistics collector in the same way as the garbage collectors, but there are only three modifiable fields.

37.4 Managing Oracle Internet Directory Garbage Collectors Logging

The following topics describe how to enable, disable and monitor the Oracle Internet Directory Garbage Collectors logging:

This section has the following topics:

- Enabling the Oracle Internet Directory Garbage Collectors Logging
- Disabling the Oracle Internet Directory Garbage Collectors Logging
- Monitoring the Oracle Internet Directory Garbage Collection Logging

37.4.1 Enabling the Oracle Internet Directory Garbage Collectors Logging

If you enable logging for garbage collectors, then the directory server writes the information into a file in the file system.

The information that are written to file includes:

- The job identifier
- A job description of the garbage collector
- The number of entries purged
- The operation status
- The time stamp
- Any errors caught

To enable logging of garbage collection information:

1. Set the `orclpurgedebug` attribute to 1, if needed. When `orclpurgedebug` is set to 1, extra debugging detail information is logged, which can be useful for troubleshooting garbage collection problems.
2. Set the `orclpurgefilename` attribute to a valid file name for the log file, for example: `oidgc001.log`.
3. Set the `orclpurgefileloc` attribute to the path name of the directory in which the log file is located, for example: `/private/mydir/oracle/ldap/log`.
4. Enable PL/SQL I/O access to the directory specified earlier. To do this, include the following in the database:

   `UTL_FILE_DIR=`*`PATH_NAME`*

   where *`PATH_NAME`* is the path you specified earlier.

   > **See Also:**
   >
   > The section on the UTL_FILE_DIR parameter type in the *Oracle Database Reference*

5. Shut down the replication server, then the Oracle Internet Directory server.
6. Restart the database.
7. Start the Oracle Internet Directory server, then the replication server.

37.4.2 Disabling the Oracle Internet Directory Garbage Collectors Logging

To disable logging of garbage collection information, set the `orclpurgedebug` attribute to `0`.

> **Note:**
>
> Even when `orclpurgedebug` is set to 0, minimal information about garbage collector operation is still logged to indicate the garbage collector's acitivites.

37.4.3 Monitoring the Oracle Internet Directory Garbage Collection Logging

The information in the garbage collection log can be useful for monitoring and troubleshooting garbage collection. You determine the location of the log by setting attributes when enabling logging.

For example, if you configured:

```
orclpurgefilename = oidgc001.log
orclpurgefileloc     = /private/mydir/oracle/ldap/log
```

when you enabled logging, then you can monitor change log garbage collection activities by reading the file `/private/mydir/oracle/ldap/log/oidgc001.log`.

The following is an example of the information logged when an administrator modified the `orclpurgenow` attribute of the change log garbage collection configuration entry:

```
Running Garbage Collector: cn=changelog purgeconfig
Starting time: 2005/03/24 11:03:23
PurgeConfig object located, Eid= 936
purge_ODSChglog: Nothing to be purged(no_work_to_do)
purge_ODSChglog: 107 chglogs successfully purged
purge_ASRChglog: Nothing to be purged(no_work_to_do)
purge_ASRChglog: 0 chglogs successfully purged
purge_ASRChglog: 0 chglogs successfully purged

Modifying Garbage Collector for at "2005/03/24 11:03:23
Garbage Collector DN recognized, rdn=cn=changelog purgeconfig
orclPurgeNow successfully retrieved.
Garbage Collector job found: jobno=21
Garbage Collector has been run
Garbage collector is updated successfully!
```

Modifying `orclpurgenow` forces the change log garbage collector to run immediately. As shown in the first paragraph, 107 change logs were purged from the `ods_chg_log` table and 0 change logs were purged from the `asr_chg_log` table. Also, the information in the second paragraph indicates successful modification of `orclpurgenow` attribute.

37.5 Configuring Time-Based Change Log Purging

You can configure time-based purging by modifying the `orclpurgetargetage` attribute of the changelog purging configuration entry.

Change log purging was described in Overview of Change Log Purging. This example configures time-based purging for 120 hours (5 days). Use an LDIF file similar to this:

```
dn: cn=changelog purgeconfig,cn=purgeconfig,cn=subconfigsubentry
changetype:modify
replace: orclpurgetargetage
orclpurgetargetage: 120
```

To apply the ldif file `mod.ldif`, type:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -f mod.ldif
```

Note:

The container for the change log garbage collector is `cn=changelog purgeconfig, cn=purgeconfig,cn=subconfigsubentry`.

See Also:

Oracle Internet Directory Configuration Schema Elements in *Reference for Oracle Identity Management*.

38
Migrating Data from Other Data Repositories

The following topics describe how to migrate data from both LDAP Version 3-compatible directories and application-specific data repositories into Oracle Internet Directory:

- Understanding Data Migration from Other Data Repositories
- Migrating Data from LDAP-Compliant Directories
- Migrating User Data from Application-Specific Repositories

38.1 Understanding Data Migration from Other Data Repositories

During an Oracle Internet Directory installation, Oracle Identity Management 14*c* Installer creates a default schema and directory information tree (DIT).

Understanding the Concepts and Architecture of Oracle Internet Directory, and Planning, Deploying and Managing Realms, describe this default DIT framework. The framework is flexible and you can modify it to suit the needs of your deployment.

In Oracle Internet Directory, the following directory elements are created by default:

- Root Oracle Context (`cn=OracleContext`): This is the container where Oracle products store enterprise-wide configuration data.
- Default identity management realm (`dc=`*`dns_domain_of_host`*`,dc=com`): This is the container under which Oracle products expect to find enterprise users and groups. It approximates the enterprise DIT structure. For example, if Oracle Internet Directory is installed on a computer whose host name is: `my_computer.us.my_company.com`, then the default identity management realm created at installation of Oracle Internet Directory would be `dc=us,dc=my_company,dc=com`. Oracle products expect to find all users under the container c`n=users,dc=us,dc=my_company,dc=com` and all groups under `cn=groups,dc=us,dc=my_company,dc=com`. In addition to creating the default identity management realm entry, the Oracle Internet Directory Configuration Assistant stores a pointer to it in the Root Oracle Context so that other Oracle Internet Directory-enabled components can bootstrap themselves.

You can change this default identity management realm to suit your deployment requirements.

38.2 Migrating Data from LDAP-Compliant Directories

If you have a directory with an already-established structure, and you want to migrate the data from that directory into the default directory structure environment, then follow the instructions in this section.

The following topics provide practical information for migrating data from an LDAP-compliant, third-party directory to Oracle Internet Directory:

- Understanding Data Migration Tools
- Migrating LDAP Data Using LDIF File and Bulk Loader
- Migrating LDAP Data Using Directory Integration Assistant Directly

- Migrating LDAP Data Using LDIF File and Directory Integration Assistant
- Migrating LDAP Data Using Directory Integration Assistant, Bulk Loader, and LDIF Files
- Migrating LDAP Data Using the Oracle Directory Integration Platform Server

38.2.1 Understanding Data Migration Tools

The following topics describe the tools that are commonly used for migrating data and also provide a feature comparison between them:

- Bulk Loader
- Directory Integration Assistant
- Features Comparison Between Bulk Loader and Directory Integration Assistant
- LDIF File

38.2.1.1 Bulk Loader

The bulk loader, `bulkload`, is a command-line tool for loading a large number of entries into a directory server. It uses Oracle SQL*Loader to load the directory entries. The `bulkload` tool expects the input file to be in LDAP Data Interchange Format (LDIF). The `bulkload` tool can validate LDIF input for referential integrity, but it cannot perform any mapping or other transformation on the data.

When no translation is required and data is very large (500,000 or more), `bulkload` is the best choice for migrating data from a third-party directory to Oracle Internet Directory. It is fast and it can validate LDIF input.

For `bulkload` syntax information and examples, see *Oracle Internet Directory Data Management Tools* in *Reference for Oracle Identity Management*.

38.2.1.2 Directory Integration Assistant

The Directory Integration Assistant, `syncProfileBootstrap`, is a command-line tool for administering the synchronization profiles scheduled by the Oracle directory integration server. An administrator can use the `syncProfileBootstrap` operation to perform the initial migration of data between a connected directory and Oracle Internet Directory when configuring the Oracle directory integration server to perform ongoing synchronization. You also use it for a one-time data migration, without ongoing synchronization.

For more information about `syncProfileBootstrap`, see Directory Bootstrapping Using syncProfileBootstrap in *Administering Oracle Directory Integration Platform*.

38.2.1.3 Features Comparison Between Bulk Loader and Directory Integration Assistant

Table 38-1 lists the features of `bulkload`, as compared with `syncProfileBootstrap`.

Table 38-1 Features of bulkload and syncProfileBootstrap

Feature	bulkload	syncProfileBootstrap
Speed	Fast	Slow
Data transfer method	SQL	LDAP

Table 38-1 (Cont.) Features of bulkload and syncProfileBootstrap

Feature	bulkload	syncProfileBootstrap
Input types accepted	LDIF file only	LDIF file, LDAP directory, tagged file, CSV file
Transforms data	No	Yes
Validates LDIF input	Yes	No

38.2.1.4 LDIF File

LDIF is the IETF-sanctioned ASCII interchange format for representing LDAP-compliant directory data as a file. All LDAP-compliant directories should have tools to export their contents into one or more LDIF files representing the DIT at the time of export.

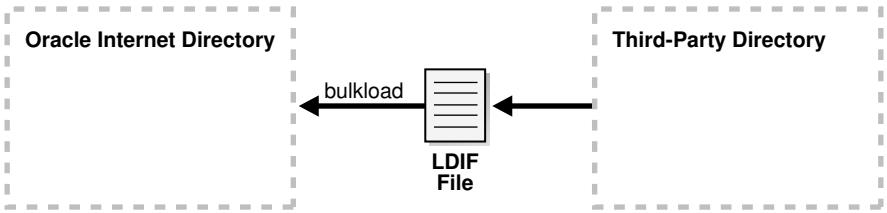
> **See Also:**
>
> RFC 2849 of the IETF, available for download at: `http://www.ietf.org`

38.2.2 Migrating LDAP Data Using LDIF File and Bulk Loader

To use this method, you must first export data from the third-party directory to an LDIF file.

See Figure 38-1.

Figure 38-1 Using an LDIF File and Bulk Loader



To migrate data to Oracle Internet Directory using an LDIF file and bulk loader:

1. Export data from the non-Oracle Internet Directory server into LDIF file format.

   See the vendor-supplied documentation for instructions. If flags or options exist for exporting data from the foreign directory, be sure to select the method that:

   - Produces LDIF output with the least amount of proprietary information included
   - Provides maximum conformance to the IETF Request for Comments 2849 of the IETF, available for download at: `http://www.ietf.org`

2. Analyze the LDIF user data for any required schema additions referenced in the LDIF data.

   Any attributes not found in the Oracle Internet Directory base schema require extension of the Oracle Internet Directory base schema before the importation of the LDIF file. Some directories may support the use of configuration files for defining extensions to their base

schema (Oracle Internet Directory does not). If you have a configuration file you can use it as a guideline for extending the base schema in Oracle Internet Directory.

3. Extend the schema in Oracle Internet Directory.

   See Managing Directory Schema for tips on how to extend the directory schema in Oracle Internet Directory. You can do this by using either Oracle Directory Services Manager or the SchemaSynch tool, which is documented in *Reference for Oracle Identity Management*.

   If you have users who are using other Oracle products, you must create users with object class `orclUserV2` and its required attributes. If you are integrating with Active Directory, you must create users with object class `orclADUser` and its required attributes. These object classes and their attributes are documented inOracle Identity Management Object Class Reference in *Reference for Oracle Identity Management*.

4. Remove any proprietary directory data from the LDIF file.

   Certain elements of the LDAP v3 standard have not yet been formalized, such as ACI attributes. As a result, various directory vendors implement ACI policy objects in ways that do not translate well across vendor installations.

   After the basic entry data has been imported from the cleaned up LDIF file to Oracle Internet Directory, you must explicitly reapply security policies in the Oracle Internet Directory environment. You can do this by using either Oracle Directory Services Manager, or command-line tools and LDIF files containing the desired ACP information.

   There may be other proprietary metadata unrelated to access control. You should remove this as well. Understanding the various IETF RFCs can help you determine which directory metadata is proprietary to a given vendor and which complies with the LDAP standards, and is thus portable by way of an LDIF file.

5. Remove operational attributes from the LDIF file.

   Four of the standard LDAP v3 operational attributes, namely, `creatorsName`, `createTimestamp`, `modifiersName`, and `modifyTimestamp` are automatically generated by Oracle Internet Directory whenever entries are created or imported. It is not possible to instantiate these values from existing directory data, for example by using LDIF file importation. Therefore you should remove these attributes from the file before attempting to import.

6. Remove incompatible userPassword attribute values from the LDIF file.

   The userPassword attribute hash algorithms supported by Oracle Internet Directory are listed in the `orclcryptoscheme` entry in Attributes of the DSE.

   The userPassword attribute hash values used by some vendor products are not compatible with Oracle Internet Directory. As a result, you must remove all lines corresponding to the `userPassword` attribute and value from the LDIF data file unless they are represented in plain text or contain no value. After importation of the LDIF data, you must manually reenter or upload hashed userPassword information separately into the directory. Be sure that the passwords comply with the Oracle Internet Directory password policies and are in clear text.

7. Run the bulkload check =TRUE" mode and determine any remaining schema violations or duplication errors.

   Before generating and loading an LDIF file, always perform a check on it by using the bulkload utility check mode. The bulkload output reports any inconsistencies in the data.

> **See Also:**
>
> The `bulkload` command-line tool reference in *Reference for Oracle Identity Management* for instructions on how to use the bulkload check mode

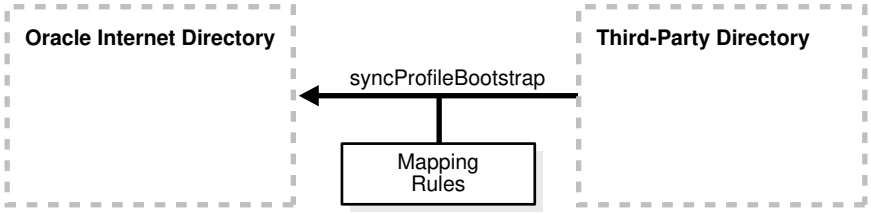
38.2.3 Migrating LDAP Data Using Directory Integration Assistant Directly

The `syncProfileBootstrap` operation can take data either directly from a third-party LDAP-compliant directory or from an LDIF file, tagged file, or CSV file. You must provide mapping rules, either as a synchronization profile or in a configuration file.

For `syncProfileBootstrap` syntax information, configuration file properties, information about input file types, and examples, see *Oracle Directory Integration Platform Tools* in *Reference for Oracle Identity Management* and Directory Bootstrapping Using syncProfileBootstrap in *Administering Oracle Directory Integration Platform*.

If you must perform mapping when migrating the data from the third-party directory to Oracle Internet Directory, and if the data is small in size, you can use `syncProfileBootstrap`. As shown in Figure 38-2, you can use the third-party directory itself as input to `syncProfileBootstrap`.

Figure 38-2 Using syncProfileBootstrap Directly

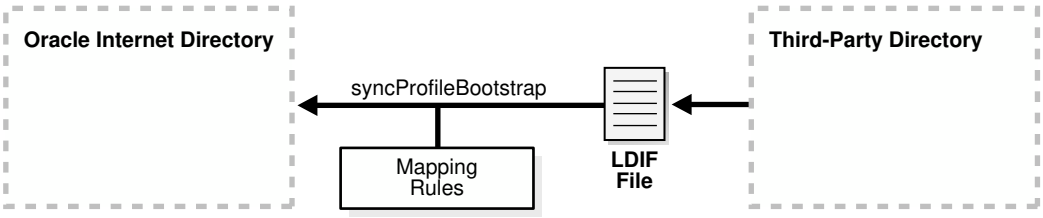


38.2.4 Migrating LDAP Data Using LDIF File and Directory Integration Assistant

If you do not have direct access to the third-party directory, you can have the administrator export the data to an LDIF file.

SeeFigure 38-3, `syncProfileBootstrap` can take its input from an LDIF file. You could also use Oracle directory integration server to migrate the data.

Figure 38-3 Using an LDIF File and syncProfileBootstrap



Whenever you use an LDIF file and bulkload to migrate data to Oracle Internet Directory, you must perform certain tasks. In this scenario, you are using a mapping file with `syncProfileBootstrap` or Oracle Directory Integration Platform, so do not have to perform all the tasks listed in "Migrating LDAP Data Using LDIF File and Bulk Loader".

The following table lists the tasks to migrate data to Oracle Internet Directory using an LDIF file and Directory Integration Assistant:

Table 38-2 List of Tasks to Migrate Data to Oracle Internet Directory using LDIF File and Directory Integration Assistant

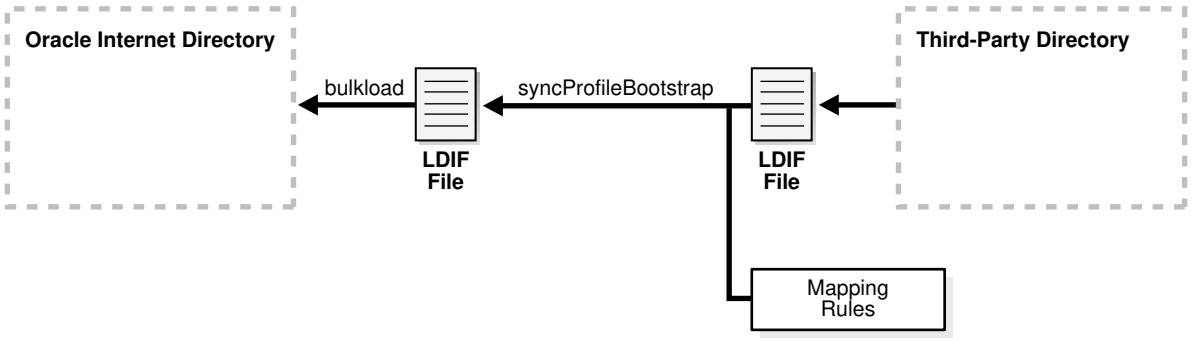
Task #	Task Description
Task 1	Export data from the non-Oracle Internet Directory Server into LDIF file format.
Task 2	Analyze the LDIF user data for any required schema additions referenced in the LDIF data.
Task 3	Extend the schema in Oracle Internet Directory.

38.2.5 Migrating LDAP Data Using Directory Integration Assistant, Bulk Loader, and LDIF Files

If you have a large amount of data and you must perform mapping on the data, you can use a combination of tools.

See Figure 38-4, you can export the data from the third-party directory to an LDIF file, then use `syncProfileBootstrap` to perform the mapping into another LDIF file, which you then load with `bulkload`.

Figure 38-4 Using syncProfileBootstrap, bulkload, and LDIF Files



The following table lists the tasks to migrate data to Oracle Internet Directory using an LDIF file, Bulk Loader and Directory Integration Assistant:

Table 38-3 List of Tasks to Migrate Data Using Directory Integration Assistant, Bulk Loader and LDIF Files

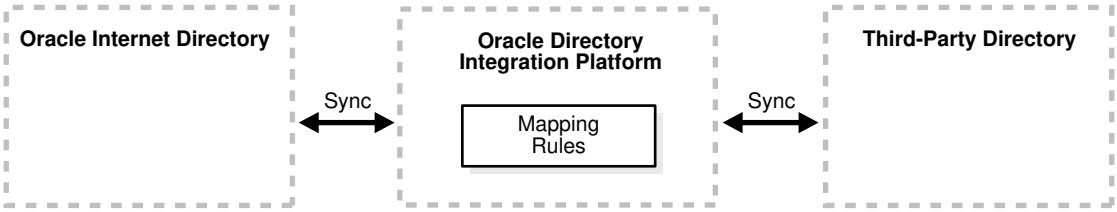
Task #	Task Description
Task 1	Export data from the non-Oracle Internet Directory server into LDIF file format.
Task 2	Analyze the LDIF user data for any required schema additions referenced in the LDIF data.
Task 3	Extend the schema in Oracle Internet Directory.

38.2.6 Migrating LDAP Data Using the Oracle Directory Integration Platform Server

Under some circumstances, an administrator might choose not to use `syncProfileBootstrap` when configuring the Oracle directory integration server. After it is configured, the Oracle directory integration server itself can migrate data from a connected directory to Oracle Internet Directory. You can also use the Oracle directory integration server for a one-time data migration.

The Oracle directory integration server enables you to configure bidirectional, ongoing integration between Oracle Internet Directory and a Third-party directory, as shown in Figure 38-5. For more information, see Understanding the Oracle Directory Synchronization Service in *Administering Oracle Directory Integration Platform*.

Figure 38-5 Using the Oracle Directory Integration Server



38.3 Migrating User Data from Application-Specific Repositories

This section given information on how to Migrate user data from an application-specific repository requires:

Migrating user data from an application-specific repository requires:

- Collecting the user data from the application-specific repository and formatting it in a way that the directory can read it
- Making that data available to the directory administrator who must then:
  - Specify where to place it in the directory
  - Import it into the directory

The following topics provide the generic instructions for enabling data migration from an application-specific repository:

- Enabling Data Migration from Application-Specific Repositories
- Reconciling Data in Application Repository with Existing Data in a Directory
- Managing Data Migration from Application-Specific Repositories

38.3.1 Enabling Data Migration from Application-Specific Repositories

To enable this migration to happen, the DSPS requires the application-specific repository to export its data to an intermediate template file. Records in this template file are not in pure LDIF; they contain substitution variables that have to do with, for example, the location in the directory where the information is finally to reside. The application leaves these variables undefined, so that you, the directory administrator can define them later on.

To convert the user data from this intermediate template file into proper LDIF, you use the OID Migration Tool (ldifmigrator). After the data is converted to LDIF, you can load it into the directory.

To summarize, migrating data from application-specific repositories involves these general steps:

1. Exporting the application-specific data as an intermediate template file.
2. You, the directory administrator, using the OID Migration Tool (ldifmigrator) to read these partial LDIF entries and convert them to pure LDIF entries based on the deployment choices
3. You, the directory administrator, loading the data, now in pure LDIF, into Oracle Internet Directory.
4. The application completing the migration process according to its own specifications.

38.3.2 Reconciling Data in Application Repository with Existing Data in a Directory

The data you are migrating from an application-specific repository may already reside in Oracle Internet Directory. If this is the case, then you can reconcile differences between the two directories by using the reconciliation feature of the OID Migration Tool (ldifmigrator).

See `ldifmigrator` command-line tool reference in *Reference for Oracle Identity Management* for information about the reconciliation feature of the OID Migration Tool.

38.3.3 Managing Data Migration from Application-Specific Repositories

The following topics describe how to create an intermediate template file and how to run the migration tool while migrating data from application-specific repositories:

- Creating an Intermediate Template File
- Running the OID Migration Tool

38.3.3.1 Creating an Intermediate Template File

The following topics describe the format and structure of the intermediate template file and also describe the attributes in a user entry:

- Intermediate Template File
- Example of User Entries in an Intermediate Template File

- Attributes in User Entries

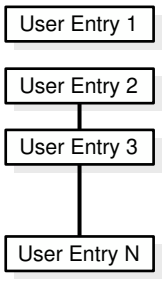
38.3.3.1.1 Intermediate Template File

Applications generating data in national languages must store that data in AL32UTF8 in the intermediate template file as specified in the IETF RFC 2849, "The LDAP Data Interchange Format (LDIF) - Technical Specification" available at http://www.ietf.org.

When generating the intermediate template file, migrating applications must list all user records sequentially with a record separator as defined in RFC 2849. The OID Migration Tool (ldifmigrator) assigns all of these users to the default identity management realm, which corresponds to the enterprise itself.

Figure 38-6 shows the overall structure of the intermediate template file containing user entries.

Figure 38-6 Structure of the Intermediate User File



The intermediate template file uses the following format to generate a valid user entry. All of the strings in **bold text** are supplied from the application-specific repository.

```
dn: cn=UserID, %s_UserContainerDN%
sn: Last_Name
orclGlobalID: GUID_for_User
%s_UserNicknameAttribute%: UserID
objectClass: inetOrgPerson
objectClass: orclUserV2
```

In this template, the strings ***%s_UserContainerDN%*** and ***%s_UserNicknameAttribute%*** are substitution variables for which the OID Migration Tool provides values. The OID Migration Tool determines these values according to deployment-specific considerations. Either the application passes the arguments to the OID Migration Tool, or the tool retrieves them from the directory.

38.3.3.1.2 Example of User Entries in an Intermediate Template File

The following intermediate template file includes user entries generated by the application-specific migration logic. In this example, all of the data listed in **bold text** is from the application-specific user repository.

```
dn: cn=jdoe, %s_UserContainerDN%
sn: Doe
%s_UserNicknameAttribute%: jdoe
objectClass: inetOrgPerson
objectClass: orclUserV2
title: Member of Technical Staff
```

```
homePhone: 415-584-5670
homePostalAddress: 234 Lez Drive$ Redwood City$ CA$ 94402


dn: cn=jsmith, %s_UserContainerDN%
sn: Smith
%s_UserNicknameAttribute%: jsmith
objectClass: inetOrgPerson
objectClass: orclUserV2
title: Member of Technical Staff
homePhone: 650-584-5670
homePostalAddress: 232 Gonzalez Drive$ San Francisco$ CA$ 94404


dn: cn=lrider, %s_UserContainerDN%
sn: Rider
%s_UserNicknameAttribute%: lrider
objectClass: inetOrgPerson
objectClass: orclUserV2
title: Senior Member of Technical Staff
homePhone: 650-584-5670
```

After all of the user data is converted to the intermediate file format, the OID Migration Tool further converts it into a proper LDIF file that can be loaded into Oracle Internet Directory.

You can find examples of intermediate template files in *$ORACLE_HOME*/ldap/schema/oid.

38.3.3.1.3 Attributes in User Entries

Each user entry has mandatory and optional attributes.

Table 38-4 lists and describes the mandatory attributes in a user entry.

Table 38-4 Mandatory Attributes in a User Entry

Attribute	Description
dn	Distinguished name of the user entry with appropriate substitution variables. The relative distinguished name of the entry MUST contain the cn attribute.
sn	Surname—that is, the last name—of the user
objectclass	Object classes the entry should minimally belong to: inetOrgPerson and orclUserV2

See Also:

- IETF Request for Comments 2798: "Definition of the inetOrgPerson LDAP Object Class," available at http://www.ietf.org, for a description of each attribute in the inetOrgPerson object class
- orclUserV2object class in *Reference for Oracle Identity Management* for more information.

38.3.3.2 Running the OID Migration Tool

After you set up the intermediate template file, the OID Migration Tool enables you to bring all pertinent data from the application-specific repository into Oracle Internet Directory. After you have migrated the data, you can update whatever portion of it is relevant to the application by synchronizing that application with Oracle Internet Directory. You synchronize by using the Oracle Directory Synchronization Service.

> **See Also:**
>
> The `ldifmigrator` command-line tool reference in *Reference for Oracle Identity Management* for instructions about using the OID Migration Tool

39
Configuring Directory Server Chaining

Directory server chaining for Oracle Internet Directory allows you to map entries that reside in external or third-party LDAP directories to part of the directory tree and then access those entries through Oracle Internet Directory, without synchronization or data migration. With server chaining, you can use the Oracle Internet Directory authorization framework when the actual identity data resides outside of Oracle Internet Directory.
This chapter includes the following sections:

- Understanding Directory Server Chaining Configuration
- Configuring Server Chaining
- Creating Server Chaining Configuration Entries
- Debugging Server Chaining
- Configuring an Active Directory Plug-in for Password Change Notification

> **Note:**
>
> In this chapter, references to Oracle Single Sign-On refer to Oracle Single Sign-On 10*g* (10.1.4.3.0) or later. References to Oracle Enterprise User Security refer to the 10*g* release only.

39.1 Understanding Directory Server Chaining Configuration

This section provide a contextual description about directory server chaining and include a list of Oracle products and other external directory servers that can be integrated with Oracle Internet Directory server chaining

This section has the following topics :

- About Oracle Internet Directory Server Chaining
- Supported External Directory Servers
- Integrating Oracle Products with Oracle Internet Directory Server Chaining
- Supported Operations for Server Chaining
- About the Role of Server Chaining in Replication

39.1.1 About Oracle Internet Directory Server Chaining

Directory server chaining was first introduced for Oracle Internet Directory 10*g* (10.1.4.0.1) and was implemented using the Java plug-in framework. You can now configure server chaining to use SSL. Server chaining does not replace Oracle Directory Integration Platform, but instead offers complementary functionality to that product.

Server chaining is different from a virtual directory such as Oracle Virtual Directory. A virtual directory is a flexible virtualization layer between multiple identity repositories and applications.

A virtual directory offers complementary services to identity synchronization and directory servers. Organizations can create consolidated logical or virtual views of data that can span multiple directories and databases.

39.1.2 Supported External Directory Servers

Oracle Internet Directory server chaining supports external directory servers:

The supported external directory servers are:

- Microsoft Active Directory

> **Note:**
>
> Oracle Internet Directory server chaining does not support Microsoft Active Directory Lightweight Directory Service (AD LDS), formerly known as ADAM.

- Oracle Directory Server Enterprise Edition (ODSEE)
- Sun Java System Directory Server (formerly known as Sun ONE or iPlanet Directory Server)
- Novell eDirectory

Oracle Internet Directory can connect with one Active Directory server, one Sun Java System Directory Server, one Novell eDirectory, or with all three directory servers.

39.1.3 Integrating Oracle Products with Oracle Internet Directory Server Chaining

The following Oracle products have been integrated with Oracle Internet Directory server chaining:

- Oracle Single Sign-On10*g* (10.1.4.3.0) or Later
- Enterprise User Security 10*g* Only

39.1.3.1 Oracle Single Sign-On10*g* (10.1.4.3.0) or Later

When server chaining is enabled, a user from the external directory can log in through Oracle Single Sign-On as if authenticated locally within Oracle Internet Directory, rather than the external repository.

39.1.3.2 Enterprise User Security 10*g* Only

Oracle Internet Directory server chaining enables you to implement Enterprise User Security 10*g* without synchronizing identity data with Oracle Internet Directory through Oracle Directory Integration Platform. Your identity data remains in the external repository and the Oracle Internet Directory data store contains only Enterprise User Security-related metadata.

With Sun Java System Directory Server as the external directory, server chaining supports password-based authentication with Enterprise User Security.

With Active Directory as the external directory, server chaining supports Kerberos- based authentication and password-based authentication with Enterprise User Security. The external users can log in to Oracle Database after the Enterprise User Security authentication setup is

completed. For further details, see Configuring an Active Directory Plug-in for Password Change Notification , which is based on Note 452385.1 on My Oracle Support (formerly MetaLink), http://metalink.oracle.com.

> **See Also:**
>
> *Oracle Database Enterprise User Security Administrator's Guide* for more information on configuring Enterprise User Security for password authentication and Kerberos authentication.

39.1.4 Supported Operations for Server Chaining

Server chaining supports the following operations:

- Bind
- Compare
- Modify
- Search

The compare, modify, and search operations can be enabled or disabled by setting configuration parameters.

When an Oracle Internet Directory client application issues an LDAP search request, Oracle Internet Directory integrates the search results from its own data and the external directories.

When an Oracle Internet Directory client application issues an LDAP bind, compare, or modify request, Oracle Internet Directory redirects the request to the external directory.

The compare operation is only supported for the `userpassword` attribute.

The attribute modification is supported in two cases:

- The external attribute has the same name as the Oracle Internet Directory attribute. This is true for most standard LDAP attributes.
- The external attribute is mapped to an Oracle Internet Directory attribute, and neither the external nor the Oracle Internet Directory attribute is an operational attribute.

> **Note:**
>
> You cannot modify an Active Directory user password from Oracle Internet Directory through server chaining.

39.1.5 About the Role of Server Chaining in Replication

If you use server chaining in a replication environment, set it up on all nodes so that the entries remain consistent across nodes.

Configure server chaining so that the mapped external directories are the same for all the replicated nodes.

39.2 Configuring Server Chaining

The following topics describe about the server chaining entries and how to customize those entries for your environment:

- About Server Chaining Entries
- Configuring Server Chaining by Using Oracle Directory Services Manager
- Configuring Server Chaining from the Command Line

39.2.1 About Server Chaining Entries

Oracle Internet Directory is shipped with disabled sample server chaining entries.

The DNs for the server chaining entries are:

- Active Directory: `cn=oidscad,cn=OID Server Chaining,cn=subconfigsubentry`
- Oracle Directory Server Enterprise Edition and Sun Java System Directory Server (formerly Sun ONE or iPlanet): `cn=oidsciplanet,cn=OID Server Chaining,cn=subconfigsubentry`
- Novell eDirectory: `cn=oidscedir,cn=OID Server Chaining,cn=subconfigsubentry`

You configure server chaining by customizing the preceding entries for your environment and then enabling them. You can perform this configuration either from the command line or by using Oracle Directory Services Manager as described later in this section.

You can add your own entry under `cn=OID Server Chaining, cn=subconfigsubentry` to configure server chaining. Here, it is mandatory to add the `orcloidscdirtype` attribute to the new entry. This attribute specifies the external directory for which you want to configure server chaining, and can have one of the following values:

- For Active directory: `ad`
- For Novell eDirectory: `edir`
- For Sun Java System Directory Server: `iplanet`

For instance, if an entry has `orcloidscdirtype=edir`, then in other words it implies that this entry is configured for connecting with eDirectory. Likewise, you can have your own set of entries.

> **Note:**
>
> Only one active entry for an external directory server type is supported.

39.2.2 Configuring Server Chaining by Using Oracle Directory Services Manager

Oracle Directory Services Manager provides a convenient interface for modifying the Oracle Internet Directory server chaining configuration entries.

To configure server chaining by using Oracle Directory Services Manager:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Advanced**.
3. Expand **Server Chaining**. Server Chaining entries appear in the left panel. Current entries include iPlanet (Oracle Directory Server Enterprise Edition and Sun Java System Directory Server) and Active Directory.
4. To modify a server chaining configuration entry, select it. The Server Chaining Management tab appears in the right pane.
5. Modify **External Host Name**, **External Port Number**, **Login User DN**, and **Login User Password** as necessary.
6. To enable server chaining authentication, modification, or search, select the corresponding checkbox.
7. Modify the other fields as necessary.
8. After modifying an external user container, group container, or login credential, verify the value by clicking **Verify User Container**, **Verify Group Container**, or **Verify Login Credential**, respectively.

   If the verification fails, examine the values you entered for errors. If the problem persists, consult the external directory administrator to verify the accuracy of the values you entered.
9. If you want to add an attribute mapping, click the **Add attribute mappings to list** icon under Attribute Mapping. To edit an existing mapping, select the mapping and click the **Edit Attribute Mapping** icon under Attribute Mapping. The New Attribute Mapping window appears. Enter the External Directory Attribute and the OID Attribute. To locate Oracle Internet Directory attribute by browsing, click **Select** then select the attribute in the Attribute Selection window.
10. Click **OK** to create the mapping or click **Cancel** to abandon it.
11. To delete a mapping, select the mapping and click the **Delete selected attribute mapping** icon. When the Delete Confirm dialog appears, click **Delete** to delete the mapping or **Cancel** to abandon deletion.
12. Click **OK** to enable the configuration changes or click **Cancel** to abandon the changes.

39.2.3 Configuring Server Chaining from the Command Line

You can configure server chaining from the command line:

1. Create an LDIF file to manually add the user and group containers. To determine the DNs for these containers, see Naming Conventions for User and Group Containers. For example, if your user search base is `cn=users,dc=us,dc=oracle,dc=com`, and the group search base is `cn=groups,dc=us,dc=oracle,dc=com`, then you would use the following entries in your LDIF file:

```
dn: cn=AD,cn=users,dc=us,dc=oracle,dc=com
cn: AD
objectclass: orclcontainer
objectclass: top

dn: cn=iPlanet,cn=users,dc=us,dc=oracle,dc=com
cn: iPlanet
objectclass: orclcontainer
objectclass: top
```

```
dn: cn=AD,cn=groups,dc=us,dc=oracle,dc=com
cn: AD
objectclass: orclcontainer
objectclass: top

dn: cn=iPlanet,cn=groups,dc=us,dc=oracle,dc=com
cn: iPlanet
objectclass: orclcontainer
objectclass: top
```

2. Use `ldapadd` and the LDIF file you just created to add the entries.

   ```
   ldapadd -p port -h host -D "binddn" -q -v -f container_ldif_file_name
   ```

3. Create another LDIF file to modify and enable the server chaining configuration entries. For example LDIF files, see Example of Configuring an Active Directory for Server Chaining and Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining. A table of attributes is provided in Creating Server Chaining Configuration Entries Attribute mapping is explained in Mapping of Oracle Internet Directory Attributes to External Directory Attributes.

4. Modify the server chaining configuration entries using the `ldapmodify` command and the LDIF file you just created. Use a command line of the form:

   ```
   ldapmodify -D "cn=orcladmin" -q -p port -h host -D "binddn" \
      -v -f entry_ldif_file_name
   ```

39.3 Creating Server Chaining Configuration Entries

The following topics describe how to configure external directory servers for server chaining either with or without SSL:

- Server Chaining Configuration Entry Attributes
- Naming Conventions for User and Group Containers
- Mapping of Oracle Internet Directory Attributes to External Directory Attributes
- Example of Configuring an Active Directory for Server Chaining
- Configuring an Active Directory for Server Chaining
- Example of Configuring an Active Directory for Server Chaining with SSL
- Configuring an Active Directory for Server Chaining with SSL
- Adding New Attributes to an Existing Active Directory Server Chaining Entry
- Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining
- Configuring Sun Java System Directory Server (iPlanet) for Server Chaining
- Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining with SSL
- Configuring Oracle Directory Server Enterprise Edition and Sun Java System Directory Server (iPlanet) for Server Chaining with SSL
- Example of Configuring an eDirectory for Server Chaining
- Example of Configuring an eDirectory for Server Chaining with SSL

39.3.1 Server Chaining Configuration Entry Attributes

This section lists and describes the server chaining configuration entry attributes.

Table 39-1 lists the configuration entry attributes for server chaining.

Table 39-1 Configuration Entry Attributes for Server Chaining

Attribute	Required	Description
`orclOIDSCExtHost`	Yes	The host name of the external directory host. This is a single value attribute.
`orclOIDSCExtPort`	Yes	The port number of the external directory host. This is a single value attribute. The default value is 3060.
`orclOIDSCExtDN`	Yes	The DN in the external directory. Server chaining binds against the external directory using this identity to perform search and modify operations. This identity must have sufficient privilege to perform the operation. This is a single value attribute.
`orclOIDSCExtPassword`	Yes	The password for the DN of the external directory. This is a single value attribute. Be sure to enable privacy mode to ensure that users cannot retrieve this attribute in clear text. See Enabling Privacy Mode of Sensitive Attributes.
`orclOIDSCExtUserContainer`	Yes	The user container in the external directory from which to perform the user search operation. This is a single value attribute.
`orclOIDSCExtGroupContainer`	Yes	The group container in the external directory from which to perform the group search operation. This is a single value attribute. If the external user container and the external group container are the same (that is, groups in the external directory server are stored in the same container as the users), this value must be the same as the value used for the user container (`orclOIDSCExtUserContainer` attribute).
`orclOIDSCTargetUserContainer`	Yes	The user container in Oracle Internet Directory in which the external users reside. For more information, see Naming Conventions for User and Group Containers.
`orclOIDSCTargetGroupContainer`	Yes	The group container in Oracle Internet Directory in which the external groups reside. For more information, see Naming Conventions for User and Group Containers.
`orclOIDSCAttrMapping`	No	Specifies each attribute mapping between the external directory and Oracle Internet Directory. For example, to map the `eMail` attribute from Active Directory to the `mail` attribute in Oracle Internet Directory, set this attribute to: `orclOIDSCAttrMapping;mail:eMail` For more information, see Mapping of Oracle Internet Directory Attributes to External Directory Attributes.
`orclOIDSCExtSearchEnabled`	Yes	External search capability. `0` = disabled (default), `1` = enabled. This is a single value attribute.
`orclOIDSCExtModifyEnabled`	Yes	External modify capability. `0` = disabled (default), `1` = enabled. This is a single value attribute.
`orclOIDSCExtAuthEnabled`	Yes	External authentication capability. `0` = disabled (default), `1` = enabled. This is a single value attribute.

Table 39-1 (Cont.) Configuration Entry Attributes for Server Chaining

Attribute	Required	Description
orclOIDSCSSLEnabled	No	SSL connection to the external directory. 0 = disabled (default), 1 = enabled. This is a single value attribute. Required if SSL is enabled.
orclOIDSCExtSSLPort	No	The SSL port number of the external directory host. This is a single value attribute.
OrclOIDSCWalletLocation	No	The filename and path of the wallet that contains the server certificate of the external directory. This is a single value attribute. Required if SSL is enabled
orclOIDSCWalletPassword	No	The wallet password. This is a single value attribute. Required if SSL is enabled
mapUIDtoADAttribute	No	Specifies the mapping of OID attribute "uid" to an attribute in Active Directory. You can map "uid" to any non-binary attributes defined in Active Directory. The default value is "name". This is a single value attribute.
showExternalGroupEntries	No	In a search against the group container: "base" - show entries with objectclass group (default), "sub" - show entries without objectclass "user" and "computer". This is a single value attribute. Applicable with Active Directory only.
showExternalUserEntries	No	In a one level search with an entry one level below the user container as the base: "base" - do not show any entry (default), "sub" - show entries in the subtree below the base of the search. This is a single value attribute. Applicable with Active Directory only.
addOrcluserv2ToADUsers	No	Add "orcluserv2" objectclass to entries that have objectclass user. 0 = disabled (default), 1 = enabled. This is a single value attribute. Applicable with Active Directory only.

39.3.2 Naming Conventions for User and Group Containers

The target user and group containers must be under the Oracle Internet Directory search base in order to work with Oracle Single Sign-On.

For user and group containers, use the following names:

- Active Directory: `cn=AD`
- Oracle Directory Server Enterprise Edition or Sun Java System Directory Server (iPlanet or Sun ONE Directory Server): `cn=iPlanet`
- Novell eDirectory: `cn=edir`

For example, if your user search base is `cn=users,dc=us,dc=oracle,dc=com`

Use the following names for the target user containers:

- Active Directory: `cn=AD,cn=users,dc=us,dc=oracle,dc=com`
- Oracle Directory Server Enterprise Edition or Sun Java System Directory Server: `cn=iPlanet,cn=users,dc=us,dc=oracle,dc=com`
- Novel eDirectory: `cn=edir,cn=users,dc=us,dc=oracle,dc=com`

Similarly, if your group search base is `cn=groups,dc=us,dc=oracle,dc=com`

Use the following names for the target group containers:

- Active Directory: `cn=AD,cn=groups,dc=us,dc=oracle,dc=com`
- Oracle Directory Server Enterprise Edition or Sun Java System Directory Server (iPlanet or Sun ONE Directory Server): `cn=iPlanet,cn=groups,dc=us,dc=oracle,dc=com`
- Novel eDirectory: `cn=edir,cn=groups,dc=us,dc=oracle,dc=com`

> **Note:**
>
> The target user and group containers exist only for the external directories. All the users and groups that appear under these nodes are populated by the external directories. Do not add entries under these containers directly from Oracle Internet Directory.
>
> If the external user container and the external group container are the same (that is, groups in the external directory server are stored in the same container as the users), the value for the group container (`orclOIDSCExtGroupContainer` attribute) must be the same as the value used for the user container (`orclOIDSCExtUserContainer` attribute).

39.3.3 Mapping of Oracle Internet Directory Attributes to External Directory Attributes

If an attribute in an external directory and an Oracle Internet Directory attribute are the same, then no mapping is required. Server chaining performs some attribute mapping by default.

The following topics describe the default attribute mapping of Oracle Internet Directory to external directories:

- Default Attribute Mapping to Active Directory
- Default Attribute Mapping to Sun Java System Directory Server
- Default Attribute Mapping to Novell eDirectory

39.3.3.1 Default Attribute Mapping to Active Directory

The following table lists the default attribute mapping of Oracle Internet Directory to Active Directory:

Table 39-2 Default Attribute Mapping to Active Directory

Oracle Internet Directory Attribute	Active Directory Attribute
`orclguid`	`objectGUID`
`uid`	`name`
`orclsamaccountname`	`samaccountname`
`krbprincipalname`	`userprincipalname`

For Active Directory server chaining, you can use the `mapUIDtoADAttribute` attribute to map `uid` to any non-binary attributes defined in Active Directory.

39.3.3.2 Default Attribute Mapping to Sun Java System Directory Server

The following table lists the default attribute mapping of Oracle Internet Directory to Sun Java System Directory Server:

Table 39-3 Default Attribute Mapping to Sun Java System Directory Server

Oracle Internet Directory Attribute	Sun Java System Directory Server Attribute
orclguid	nsuniqueid
authpassword	userpassword
krbprincipalname	mail

39.3.3.3 Default Attribute Mapping to Novell eDirectory

The following table lists the default attribute mapping of Oracle Internet Directory to Novell eDirectory:

Table 39-4 Default Attribute Mapping to Novell eDirectory

Oracle Internet Directory Attribute	Novell eDirectory Attribute
orclguid	guid
orclsamaccountname	uid
krbprincipalname	mail

The following objects cannot be mapped:

- Operational attributes
- Object classes
- Oracle Internet Directory- specific attributes. These attributes typically have names starting with `orcl`.

39.3.4 Example of Configuring an Active Directory for Server Chaining

The following example shows server chaining configured to use the Active Directory server `dlin-pc9.us.example.com`, port `3060`, as its external directory store. The SSL capability has been enabled.

All the attributes are explained in Table 39-1.

```
cn=oidscad,cn=OID Server Chaining,cn= subconfigsubentry
orclOIDSCExtHost: dlin-pc9.us.example.com
orclOIDSCExtPort: 3060
orclOIDSCExtDN: cn=administrator,cn=users,dc=oidvd,dc=com
orclOIDSCExtPassword: *******
orclOIDSCExtUserContainer: cn=users,dc=oidvd,dc=com
orclOIDSCTargetUserContainer: cn=AD,cn=users,dc=us,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=AD,cn=groups,dc=us,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
```

```
orclOIDSCExtAuthEnabled: 1
orclOIDSCAttrMapping;description: title
orcloidscsslenabled: 0
```

39.3.5 Configuring an Active Directory for Server Chaining

The following example is the LDIF file used to modify the configuration entry:

The following example is the LDIF file used to modify the configuration entry:

```
dn: cn=oidscad,cn=oid server chaining,cn=subconfigsubentry
changetype: modify
replace: orcloidscextdn
orcloidscextdn: cn=administrator,cn=users,dc=oidvd,dc=com
-
replace: orcloidscextpassword
orcloidscextpassword: password
-
replace: orcloidscexthost
orcloidscexthost: dlin-pc9.us.example.com
-
replace: orcloidscextport
orcloidscextport: 3060
-
replace: orcloidsctargetusercontainer
orcloidsctargetusercontainer: cn=AD,cn=users,dc=us,dc=oracle,dc=com
-
replace: orcloidsctargetgroupcontainer
orcloidsctargetgroupcontainer: cn=AD,cn=groups,dc=us,dc=oracle,dc=com
-
replace: orcloidscextusercontainer
orcloidscextusercontainer: cn=users,dc=dlin,dc=net
-
replace: orcloidscextgroupcontainer
orcloidscextgroupcontainer: cn=users,dc=dlin,dc=net
-
replace: orcloidscextsearchenabled
orcloidscextsearchenabled: 1
-
replace: orcloidscextmodifyenabled
orcloidscextmodifyenabled: 1
-
replace: orcloidscextauthenabled
orcloidscextauthenabled: 1
-
replace: orcloidscsslenabled
orcloidscsslenabled:1
```

39.3.6 Example of Configuring an Active Directory for Server Chaining with SSL

The following example shows server chaining configured to use the Active Directory server.

The following example shows server chaining configured to use the Active Directory server ad.example.com, SSL port 3133, and the wallet located at `/adwallet/ewallet.p12`.

```
cn=oidscad,cn=OID Server Chaining,cn= subconfigsubentry
orclOIDSCExtHost: ad.example.com
orclOIDSCExtPort: 3060
orclOIDSCExtDN: cn=administrator,cn=users,dc=oidvd,dc=com
```

```
orclOIDSCExtPassword: *******
orclOIDSCExtUserContainer: cn=users,dc=oidvd,dc=com
orclOIDSCTargetUserContainer: cn=AD,cn=users,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=AD,cn=groups,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
orclOIDSCExtAuthEnabled: 1
orclOIDSCSSLEnabled: 1
orclOIDSCExtSSLPort: 3133
orclOIDSCWalletLocation: /adwallet/ewallet.p12
orclOIDSCWalletPassword: ********
```

39.3.7 Configuring an Active Directory for Server Chaining with SSL

You can configure an Active Directory for server chaining with SSL from the command line.

Perform the following steps:

1. Configure Active Directory server chaining without SSL, as described in the previous section.
2. Create an LDIF file like the following to enable SSL connection to the external directory. Replace the values of `orcloidscextsslport`, `orcloidscwalletlocation` and `orcloidscwalletpassword` with values that match the actual Active Directory server:

   ```
   dn: cn=oidscad,cn=oid server chaining,cn=subconfigsubentry
   changetype: modify
   replace: orcloidscsslenabled
   orcloidscsslenabled:1
   -
   replace: orcloidscextsslport
   orcloidscextsslport: 3133
   -
   replace: orcloidscwalletlocation
   orcloidscwalletlocation: /adwallet/ewallet.p12
   -
   replace: orcloidscwalletpassword
   orcloidscwalletpassword: passw0rd
   ```

3. To apply the changes, use a command line such as

   ```
   ldapmodify -p OID_port -h OID_host -D "cn=orcladmin" -q -v -f ldif_file_name
   ```

39.3.8 Adding New Attributes to an Existing Active Directory Server Chaining Entry

To add the attributes to an existing Active Directory server chaining entry, modify the LDIF file with the appropriate values.

The attributes `mapUIDtoADAttribute`, `showExternalGroupEntries`, `showExternalUserEntries`, and `addOrcluserv2ToADUsers` are now supported.

To add these attributes to an existing Active Directory server chaining entry, modify the following LDIF file with the appropriate values:

```
dn: cn=oidscad,cn=oid server chaining,cn=subconfigsubentry
changetype: modify
replace: mapUIDtoADAttribute
mapUIDtoADAttribute: name
-
replace: showExternalGroupEntries
```

```
showExternalGroupEntries: base
-
replace: showExternalUserEntries
showExternalUserEntries: base
-
replace: addOrcluserv2ToADUsers
addOrcluserv2ToADUsers: 0
```

Use a command line such as

```
ldapmodify -p OID_port -h OID_host -D "cn=orcladmin" -q -v -f ldif_file_name
```

to modify the configuration entry.

39.3.9 Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining

The following example shows server chaining configured to use the Sun Java System Directory Server `dlin-pc10.us.example.com`, port `103060`, as its external directory store.

All the attributes are explained in Table 39-1.

```
cn=oidsciplanet,cn=OID Server Chaining,cn=subconfigsubentry
orclOIDSCExtHost: dlin-pc10.us.example.com
orclOIDSCExtPort: 10389
orclOIDSCExtDN: cn=directory manager
orclOIDSCExtPassword: ********
orclOIDSCExtUserContainer: ou=people,dc=example,dc=com
orclOIDSCExtGroupContainer: ou=groups,dc=example,dc=com
orclOIDSCTargetUserContainer: cn=iPlanet,cn=users,dc=us,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=iPlanet,cn=groups,dc=us,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
orclOIDSCExtAuthEnabled: 1
orclOIDSCSSLEnabled:0
```

39.3.10 Configuring Sun Java System Directory Server (iPlanet) for Server Chaining

The following example is the LDIF file used to modify the configuration entry:

The following example is the LDIF file used to modify the configuration entry:

```
dn: cn=oidsciplanet,cn=oid server chaining,cn=subconfigsubentry
changetype: modify
replace: orcloidscextdn
orcloidscextdn: cn=directory manager
-
replace: orcloidscextpassword
orcloidscextpassword: password
-
replace: orcloidscexthost
orcloidscexthost: dlin-pc10.us.example.com
-
replace: orcloidscextport
orcloidscextport: 10389
-
replace: orcloidsctargetusercontainer
orcloidsctargetusercontainer: cn=iplanet,cn=users,dc=us,dc=oracle,dc=com
```

```
-
replace: orcloidsctargetgroupcontainer
orcloidsctargetgroupcontainer: cn=iplanet,cn=groups,dc=us,dc=oracle,dc=com
-
replace: orcloidscextusercontainer
orcloidscextusercontainer: ou=people,dc=example,dc=com
-
replace: orcloidscextgroupcontainer
orcloidscextgroupcontainer: ou=groups,dc=example,dc=com
-
replace: orcloidscextsearchenabled
orcloidscextsearchenabled: 1
-
replace: orcloidscextmodifyenabled
orcloidscextmodifyenabled: 1
-
replace: orcloidscextauthenabled
orcloidscextauthenabled: 1
```

39.3.11 Example of Configuring Sun Java System Directory Server (iPlanet) for Server Chaining with SSL

This example shows how to configure sun java system directory server for server chaining with SSL.

The following example shows server chaining configured to use the Sun Java System Directory Server sunone.example.com, SSL port 10636, and the wallet located at /ipwallet/ewallet.p12.

```
cn=oidsciplanet,cn=OID Server Chaining,cn=subconfigsubentry
orclOIDSCExtHost: sunone.example.com
orclOIDSCExtPort: 10389
orclOIDSCExtDN: cn=directory manager
orclOIDSCExtPassword: ********
orclOIDSCExtUserContainer: ou=people,dc=example,dc=com
orclOIDSCExtGroupContainer: ou=groups,dc=example,dc=com
orclOIDSCTargetUserContainer: cn=iPlanet,cn=users,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=iPlanet,cn=groups,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
orclOIDSCExtAuthEnabled: 1
orclOIDSCSSLEnabled: 1
orclOIDSCExtSSLPort: 10636
orclOIDSCWalletLocation: /ipwallet/ewallet.p12
orclOIDSCWalletPassword: ********
```

39.3.12 Configuring Oracle Directory Server Enterprise Edition and Sun Java System Directory Server (iPlanet) for Server Chaining with SSL

You can configure Oracle Directory server Enterprise and Sun System directory server from the command line.

To configure server chaining with SSL from the command line:

1. Configure server chaining without SSL, as described in the previous section.
2. Create the following LDIF file to enable SSL connection to the external directory. Replace the values of `orcloidscextsslport`, `orcloidscwalletlocation` and

`orcloidscwalletpassword` with values that match the actual Oracle Directory Server Enterprise Edition/Sun Java System Directory Server.

```
dn: cn=oidsciplanet,cn=oid server chaining,cn=subconfigsubentry
changetype: modify
replace: orcloidscsslenabled
orcloidscsslenabled:1
-
replace: orcloidscextsslport
orcloidscextsslport: 10636
-
replace: orcloidscwalletlocation
orcloidscwalletlocation: /ipwallet/ewallet.p12
-
replace: orcloidscwalletpassword
orcloidscwalletpassword: passw0rd
```

3. Execute a command such as

   ```
   ldapmodify -p OID_port -h OID_host -D "cn=orcladmin" -q -v -f ldif_file_name
   ```

   to modify the configuration entry.

39.3.13 Example of Configuring an eDirectory for Server Chaining

This section shows an example for configuring an eDirectory for server chaining.

A sample eDirectory configuration looks like this:

```
cn=oidscedir,cn=OID Server Chaining,cn=subconfigsubentry
orclOIDSCExtHost: edirhost.domain.com
orclOIDSCExtPort: 3060
orclOIDSCExtDN: cn=admin,o=domain
orclOIDSCExtPassword: ********
orclOIDSCExtUserContainer: ou=users,o=domain
orclOIDSCExtGroupContainer: ou=groups,o=domain
orclOIDSCTargetUserContainer: cn=edir,cn=users,dc=us,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=edir,cn=groups,dc=us,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
orclOIDSCExtAuthEnabled: 1
orclOIDSCSSLEnabled:0
```

39.3.14 Example of Configuring an eDirectory for Server Chaining with SSL

A sample edirectory configuration with SSL looks like this:

A sample edirectory configuration with SSL looks like this:

```
cn=oidscedir,cn=OID Server Chaining,cn=subconfigsubentry
orclOIDSCExtHost: edirhost.domain.com
orclOIDSCExtPort: 3060
orclOIDSCExtDN: cn=admin,o=domain
orclOIDSCExtPassword: ********
orclOIDSCExtUserContainer: ou=users,o=domain
orclOIDSCExtGroupContainer: ou=groups,o=domain
orclOIDSCTargetUserContainer: cn=edir,cn=users,dc=us,dc=oracle,dc=com
orclOIDSCTargetGroupContainer: cn=edir,cn=groups,dc=us,dc=oracle,dc=com
orclOIDSCExtSearchEnabled: 1
orclOIDSCExtModifyEnabled: 1
orclOIDSCExtAuthEnabled: 1
orclOIDSCSSLEnabled: 1
```

```
orclOIDSCExtSSLPort: 3133
orclOIDSCWalletLocation: /edir/ewallet.p12
orclOIDSCWalletPassword: ********
```

39.4 Debugging Server Chaining

This section describes the procedure to debug server chaining.

To debug server chaining:

1. Set the Oracle Internet Directory server debug logging level, as described in Managing Logging Using Fusion Middleware Control or Managing Logging from the Command Line. Use the logging level value 402653184. This value enables logging of all messages related to the Java plug-in framework.
2. Modify the Oracle Internet Directory server chaining debugging settings. For both `cn=oidscad,cn=oid server chaining,cn=subconfigsubentry` and `cn=oidsciplanet,cn=oid server chaining, cn=subconfigsubentry`. set the attribute `orcloidscDebugEnabled` to `1`.

   For example, to set `orcloidscDebugEnabled` to `1` in `cn=oidscad,cn=oid server chaining,cn=subconfigsubentry`, you would type:

   ```
   $ORACLE_HOME/bin/ldapmodify -h host -p port -D cn=orcladmin -q -f file
   ```

   where *file* contains:

   ```
   dn: cn=oidscad,cn=oid server chaining,cn=subconfigsubentry
   changetype: modify
   replace: orcloidscDebugEnabled
   orcloidscDebugEnabled: 1
   ```

> **See Also:**
>
> Java plug-in debugging and logging information in the Java Plug-ins for User Provisioning in *Application Developer's Guide for Oracle Identity Management*.

39.5 Configuring an Active Directory Plug-in for Password Change Notification

When you use Enterprise User Security 14*c* with Server Chaining, a hash password is required in order to authenticate users. This section describes how to install a plug-in the Microsoft Active Directory (AD) server so that this hash password is available to users accessed through Oracle Internet Directory. Customers planning to configure Enterprise User Security to work with users accessed through Server Chaining must configure this feature.

To configure an active directory plug-in for password change notification:

1. In Active Directory, create an attribute called `orclCommonAttribute` to store the hash password. Use a command line such as:

   ```
   ldapadd -p AD_Port -h AD_host -D "AD_administrator_DN" -w AD_administrator_password -
   v -f orclca.ldif
   ```

Use an orclca.ldif file similar to the following example. Replace `DC=bill,DC=com` with the actual Active Directory domain name and choose an appropriate `attributeID`.

```
dn: cn=orclcommonattribute,CN=Schema,CN=Configuration,DC=bill,DC=com
objectClass: top
objectClass: attributeSchema
cn: orclcommonattribute
distinguishedName: CN=orclcommonattribute,CN=Schema,CN=Configuration,DC=bill,DC=com
instanceType: 4
uSNCreated: 16632
attributeID: 1.9.9.9.9.9.9.9.9
attributeSyntax: 2.5.5.3
isSingleValued: TRUE
uSNChanged: 16632
showInAdvancedViewOnly: TRUE
adminDisplayName: orclCommonAttribute
oMSyntax: 27
lDAPDisplayName: orclCommonAttribute
name: orclcommonattribute
objectCategory: CN=Attribute-Schema,CN=Schema,CN=Configuration,DC=bill,DC=com
```

2. Associate the attribute with the user objectclass. Use a command line such as:

```
ldapadd -p AD_Port -h AD_host -D "AD_administrator_DN" -w AD_administrator_password -
v -f user.ldif
```

In the following file, user.ldif, replace D`C=bill,DC=com` with the actual Active Directory domain name.

```
dn: CN=User,CN=Schema,CN=Configuration,DC=bill,DC=com
changetype: modify
add: mayConatin
mayContain: orclCommonAttribute
```

It might take Active Directory a few minutes to refresh the schema.

3. Install the password change notification plug-in, as follows:
   a. Copy:

      `%ORACLE_HOME%\ovd\eus\win64\oidpwdcn.dll` (for Windows 64 bit)

      Or,

      `%ORACLE_HOME%\ovd\eus\win\oidpwdcn.dll` (for Windows 32 bit)

      to the Active Directory `WINDOWS\system32` folder.

   b. Use `regedt32` to modify the registry. In the line:`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Notification Packages`, add `oidpwdcn` to the end. It should look like the following:

      ```
      RASSFM
      KDCSVC
      WDIGEST
      scecli
      oidpwdcn
      ```

   c. Restart Active Directory.

   d. Verify that the plug-in is installed properly by resetting the password of a user. The `orclCommonAttribute` should contain the hash password value.

4. Reset the password for all the Active Directory users so that the password verifier is present for all the users.

40

Managing DIT Masking

The following topics describe DIT masking with Oracle Internet Directory, including how to configure the attributes required for DIT masking:

- About DIT Masking
- DIT Masking Configuration Attributes
- Configuring DIT Masking

40.1 About DIT Masking

DIT masking is the restriction of the DIT content that is exposed in an Oracle Internet Directory server instance.

DIT masking restricts access by all users except the super user, `cn=orcladmin`. Typically, you use masking to prevent some users from seeing certain portions of the DIT, based on which instance of the Oracle Internet Directory server they connect to. Typical use cases for presenting different views of the DIT include test vs. production and internal vs. external users.

You can also disallow access to one or more containers from the entire directory, as opposed to hiding the containers from a specific Oracle Internet Directory server instance.

You could also restrict a user's view of the DIT by using Oracle Virtual Directory, but DIT masking has far less performance and administrative overhead.

40.2 DIT Masking Configuration Attributes

By default, no masking is configured. You use the following configuration attributes of the instance-specific configuration entry to configure masking.

Table 40-1 Masking Configuration Attributes

Attribute	Description
`orclMaskRealm`	Contains the DIT subtrees that are exposed in an instance. The DN configured and its children are visible in the instance. Other entries in the DIT are masked (hidden) for all LDAP operations.
`orclMaskRealm;disallowed`	Contains DIT subtrees that are hidden in a container for an entire directory.
`orclMaskFilter`	Filters the entries exposed in the instance. Entries matching the filter criteria are exposed. Other entries are hidden for all LDAP operations.

You modify these attributes in the same way as other attributes of the instance-specific configuration entry. See Setting System Configuration Attributes by Using ldapmodify .

40.3 Configuring DIT Masking

Masking is useful in scenarios where the administrator wants to selectively expose or hide the entries present in the directory.

The following examples illustrate this use case.

- Restricting Access by Container Name
- Restricting Access by Entry Data
- Disallowing Access to Containers from the Entire Directory

40.3.1 Restricting Access by Container Name

This example ensures that only the entries in the configured containers public and external are seen through this instance.

Consider a DIT setup with the following hierarchy:

```
cn=internal,o=oracle
cn=external,o=oracle
cn=public,o=oracle
```

The internal container contains entries internal to the organization and should have limited access. The `external` and `public` container contains data about external users and some public information that is accessible to all. An administrator wants to ensure that only the `external` and `public` data is available outside of the organization firewall. This can be achieved through masking. Create an Oracle Internet Directory instance, such as `oid2`, that runs on a port exposed through the firewall. To ensure applications and users connecting to this port see only publicly accessible content, create masking realms in `cn=oid2` with `ldapmodify`, using the following LDIF file:

```
dn: cn=oid2,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclmaskrealm
orclmaskrealm: cn=external,o=oracle
orclmaskrealm: cn=public,o=oracle
```

This example ensures that only the entries in the configured containers public and external are seen through this instance. Applications and users connecting to this instance cannot see the Internal container and its entries.

40.3.2 Restricting Access by Entry Data

Another use case is restricting entries based on the data stored in them. An organization might have data about employees, contract workers and temp workers. A user lookup application such as an email client looks up data on the directory server to find out email addresses. An administrator wants to hide temp workers' information and only expose employees and contractor workers in the instance, say `cn=oid2`, that is accessed by the email client.

This can be done by configuring masking filters with `ldapmodify`, using the following LDIF file:

```
dn: cn=oid2,cn=osdldapd,cn=subconfigsubentry
changetype: modify
add: orclmaskfilter
orclmaskfilter: (usertype=employee)
orclmaskfilter: (usertype=contract)
```

This example ensures that entries with `usertype=employee` or `usertype=contract` are exposed and others are not exposed.

40.3.3 Disallowing Access to Containers from the Entire Directory

This example ensures that entries in the `internal` container are not accessible to users other than super user.

Consider a DIT setup with the following container:

```
cn=internal,o=oracle
```

You can disallow access to this container from the entire directory (as opposed to hiding the container from a specific instance) with `ldapmodify`, using the following LDIF file:

```
dn: cn=dsaconfig,cn=configsets,cn=oracle internet directory
changetype: modify
add: orclmaskrealm;disallowed
orclmaskrealm;disallowed: cn=internal,o=oracle
```

This example ensures that entries in the `internal` container are not accessible to users other than super user, `cn=orcladmin`, for all LDAP operations.

Part V

Advanced Administration: Directory Replication

This part provides detailed discussions of replication and high availability and how to plan and manage them.

This section contains the following chapters:

- Setting Up Replication
- Setting Up Replication Failover
- Managing Replication Configuration Attributes
- Managing and Monitoring Replication

41

Setting Up Replication

Understand how to perform replication for Oracle Internet Directory and how to set up LDAP-based replication and multi-master replication with fan-out.
Replication is the process of copying and maintaining the same naming contexts on multiple directory servers. It can improve performance by providing more servers to handle queries and by bringing the data closer to the client. It improves reliability by eliminating risks associated with a single point of failure.

Before reading this chapter, see Understanding Oracle Internet Directory Replication for an introduction to basic replication concepts.

> **See Also:**
>
> Transport Mechanism: LDAP.
>
> Oracle Internet Directory High Availability in *High Availability Guide* for information on setting up replication in high availability configurations.

This section contains the following topics:

- Introduction to Setting Up Replication
- Testing Replication by Using Oracle Directory Services Manager
- Setting Up a LDAP-Based Replication by Using the Command Line
- Scenario: Setting Up a Multimaster Replication Group with Fan-Out

> **Note:**
>
> All references to Oracle Single Sign-On or Oracle Delegated Administration Services in this chapter refer to Oracle Single Sign-On 10g (10.1.4.3.0) or later and *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager* in 12c Release 2 (12.2.1.3.0) or later.

41.1 Introduction to Setting Up Replication

This section provides an overview of how to set up replication.

If you are unfamiliar with basic replication concepts, see Understanding Oracle Internet Directory Replication before reading this introduction.

This introduction contains the following topics:

- Replication Transport Mechanisms
- Replication Setup Methods

- Bootstrap Rules for Replication
- The Replication Agreement
- Other Replication Configuration Attributes
- Replication Process and Architecture
- Rules for Configuring LDAP-Based Replication
- Replication Security
- LDAP Replication Filtering for Partial Replication

41.1.1 Replication Transport Mechanisms

Oracle Internet Directory supports LDAP-based replication transport mechanisms. It uses the industry-standard Lightweight Directory Access Protocol Version 3.

You can set up LDAP-based replication in one-way, two-way, and multimaster configurations. This is the recommended protocol for most environments.

41.1.2 Replication Setup Methods

This section provides information on replication setup methods.

To setup Oracle Internet Directory replication, use one of these methods:

- Command-line Tools to Setup and Modify Replication
- Database Copy Procedure to Replicate from an Existing Host

> **Note:**
>
> You must use Command-line Tools to Setup and Modify Replication (`ldifwrite` and `bulkload`) for the following scenario:
>
> - Replication for a directory that has more than 100,000 entries

41.1.2.1 Command-line Tools to Setup and Modify Replication

You can use command-line tools to set up LDAP-based replication.

Command-line setup of LDAP-based replication is described in Setting Up a LDAP-Based Replication by Using the Command Line.

When setting up replication from the command line, you use the `oidctl` command for stopping and starting the replication server. You use bulk tools for backing up data and loading it to other nodes. You use LDAP tools for a few operations.

> **Note:**
>
> If you start the replication server by using the command line, then you must stop it by using the command line.

Optionally, you can use the bootstrap capability of the replication server for the initial data migration.

You use the Replication Environment Management Tool, `remtool`, to perform various replication-related tasks, including:

- Setting up a replication group
- Adding and deleting replicas
- Managing the directory replication group
- Modifying or resetting the replication Bind DN password
- Modifying the database replication user REPADMIN password
- Displaying various errors and status information for change log propagation
- Tracking replication progress in a directory replication group.

> **See Also:**
>
> The `remtool` command-line tool reference in *Reference for Oracle Identity Management* for more information about the Replication Environment Management Tool

41.1.2.2 Database Copy Procedure to Replicate from an Existing Host

It is possible to set up replication on a new host by copying the Oracle Database from an existing host. This is a complex procedure that is not recommended for most environments. The procedure is described in Adding a Directory Node by Using the Database Copy Procedure.

41.1.3 Bootstrap Rules for Replication

You can use the bootstrap capability of the replication server for the initial data migration.

You set the bootstrap flag by setting the attribute `orclreplicastate` to `0` under the `replicadn`.

> **See Also:**
>
> - Setting Up an LDAP-Based Replica by Using Automatic Bootstrapping
> - LDAP Replica States

When a replica is in bootstrap mode, the supplier node must be in on line mode (`orclreplicastate=1`). Do not set the supplier and consumer to bootstrap at same time.

Bootstrap cannot be used for initial data migration on a node that has more than one supplier. Therefore, bootstrap is limited to the following types of replica:

- A leaf replica, that is, one that has no consumer replicas.
- A primary replica in a two-node LDAP-based multi-master replication agreement, provided it has no two-way fanout replicas.

- A non-primary replica in an LDAP-based multimaster replication agreement with two or more nodes, provided it has no two-way fanout replicas.
- A fanout replica with only one supplier.

If you set the bootstrap flag (`orclreplicastate=0` under `replicadn`) of any other replica, the replication server throws one of these messages:

```
bootstrapping against non-leaf replica is not allowed
bootstrap against master replica is not allowed
```

> **Note:**
>
> Disable referential integrity during the replication bootstrapping process. If referential integrity is enabled, bootstrapping fails.

41.1.4 The Replication Agreement

When you set up replication, you create a container called a replication agreement in the DIT on each of the participating hosts.

The attributes of the replication agreement entry are described in Understanding Replication Agreement Entry.

The replication agreement also contains the replication contexts. These are discussed in more detail in LDAP Replication Filtering for Partial Replication. The attributes are described in Understanding Replication Agreement Entry.

41.1.5 Other Replication Configuration Attributes

In addition to the replication agreement entry, the DIT includes several other entries that contain attributes that control replication.

The entries and their attributes are described in Managing Replication Configuration Attributes. Once you have set up replication, you can manage these attributes.

You can modify replication attributes by using LDAP tools. These methods are described in Managing and Monitoring Replication.

41.1.6 Replication Process and Architecture

In this section you can find reference information about replication architecture and the replication process.

See How Replication Works for detailed information about replication architecture and the replication process.

41.1.7 Rules for Configuring LDAP-Based Replication

Learn about the rules that apply to LDAP-based replication.

The rules that apply to LDAP-based replication are:

- If you have multiple Oracle Internet Directory instances that use the same Oracle Database, only one of the instances can be set up for replication.

- LDAP Multimaster replication is not backward compatible. It is only supported between replicas that are running 12c Release 2.
- For either multimaster replication or two-way fan-out replication, all nodes must be running the same release of Oracle Internet Directory. Therefore, you must turn off replication while performing rolling upgrades.
- You can add a one-way fan-out replica that is running a newer release than its supplier. For example, in Figure 6-5, Node F can be running a newer release than the other nodes.
- In general, do not replicate changes generated on a newer version of Oracle Internet Directory to a node that has not yet upgraded to that version. If you do, the changes can contain information that the earlier version cannot properly interpret.
- More specifically, if you add a new Oracle Internet Directory node to an existing DRG as a one-way fan-out replica, there is no need to upgrade other nodes of DRG. For all other types of replication, first upgrade the nodes in the DRG before adding the new node. This is true whether the existing nodes are running a 10*g* release or a previous 11*g* release. For example, in Figure 6-5, before adding an Oracle Internet Directory 14*c* Release (14.1.2.1.0) node as Node E, you must first update Nodes A, B, C and G to 14*c* Release (14.1.2.1.0).
- In LDAP-based replication, only the naming contexts listed in the `namingcontexts` attribute of the root DSE can be replicated to the consumer.

> **See Also:**
>
> The discussion of namingcontexts in:
>
> – Filtering of Naming Contexts in LDAP Replication
>
> – Implications of LDAP-Based Partial Replication

- The supplier of an LDAP-based replica can be a master node that is not a member of any replication group, a member of a multimaster replication group, or another LDAP-based replica.

> **See Also:**
>
> Configuring Oracle Internet Directory in *Installing and Configuring Oracle Internet Directory* for instructions on installing Oracle Internet Directory.

- An LDAP-based replica can be a consumer for another LDAP-based replica. That consumer is then called a fan-out replica.

> **Note:**
>
> Make sure the schemas are synchronized. Otherwise, the replication server might not be able to apply changes to the consumer replica.

- The new consumer node must be empty. That is, Oracle Internet Directory must be newly installed.

41.1.8 Replication Procedure for a Mixed Deployment of 10g and 11gR1 Nodes

Consider a deployment with a combination of Oracle Internet Directory 10g nodes and 11gR1 nodes.

For example:

- Two 10g nodes with a supplier node and consumer node
- Two new 11gR1 nodes (for example, nodes 1 and 2) with LDAP multimaster replication

Replication must be setup as follows:

1. Setup LDAP one-way replication from one of the 10g nodes to 11gR1 node 1.
2. Setup replication bootstrap (if less than 100,000 entries) and then start the replication server on the 11gR1 node 1.
3. When the bootstrap is complete, setup LDAP multimaster replication between 11gR1 node 1 and node 2.
4. Setup the replication bootstrap on 11gR1 node 2 and then start the replication server.

Most important, replication from the 10g node to an 11gR1 node must be setup before replication between the 11gR1 nodes is setup.

41.1.9 Replication Security

You can ensure security in replication by using authentication and SSL encryption mechanisms.

This section contains these topics:

- Authentication of the Directory Replication Server
- Use of SSL Encryption in Oracle Internet Directory Replication

41.1.9.1 Authentication of the Directory Replication Server

Authentication is the process by which the Oracle directory replication server establishes the true identity of itself when connecting to the directory server. It occurs when an LDAP session is established by means of an ldapbind operation.

It is important that the directory replication server be properly authenticated before it is allowed access to the directory.

The directory replication server uses a unique identity and a password to authenticate with the directory server. The identity of the directory replication server is of the form `cn=replication dn,orclreplicaid=`*`unique_identifier_of_node`*`,cn=replication configuration`.

When it starts, the directory replication server reads its identity and password from an Oracle Internet Directory secure wallet, and uses these credentials for authentication. If you want to change the password for the replication bind DN, then you must use the `-chgpwd`, `-presetpwd`, or `-pchgwalpwd` option of the Replication Environment Management Tool. The wallet for replication identity is located at *$DOMAIN_HOME*`/config/fmwconfig/components/OID/admin/oidpwdr`*`Oracle_SID`*.

> **See Also:**
>
> The `remtool` command-line tool reference in *Reference for Oracle Identity Management*.

> **Note:**
>
> In earlier releases, the replication server required the directory server to allow anonymous bind. The replication server no longer requires that.

41.1.9.2 Use of SSL Encryption in Oracle Internet Directory Replication

You can deploy Oracle Internet Directory replication with or without SSL. The replication server automatically detects if it is binding to the SSL port of an Oracle Internet Directory instance. If so, it automatically works on top of the Secure Sockets Layer.

To configure LDAP-based replication to use SSL encryption, specify the port number of the SSL port in the `orclReplicaURI` attribute, which contains the supplier contact information.

> **Note:**
>
> Replication server can communicate over an SSL port that is configured for one-way or two-way authentication. No-auth mode of SSL is disabled out-of-box in Oracle Internet Directory. To enable no-auth mode of SSL, anonymous cipher should be configured. See Configuring ODSM Connection with SSL Enabled for the LDIF file configuration.

> **See Also:**
>
> Configuring Secure Sockets Layer (SSL).

A sample replication entry with SSL two-way configuration for supplier node:

```
dn: orclreplicaid=myhost1_repl1,cn=replication configuration
orclreplicauri:ldap://myhost1:3131/
orclreplicauri;wallet:<location of auto-login wallet>
orclreplicauri;authmode:64
orclreplicatype: 0
orclreplicasecondaryuri:ldap://myhost1:3131/
objectclass:top
objectclass:orclreplicasubentry
orclreplicaid:myhost1_repl1
```

where `orclreplicauri;authmode` has the SSL authmode(values are 1/32/64 for no-auth/one-way/two-way authentication)

While bringing up `oidrepld` in one-way or two-way authentication mode, we need to pass value for parameters in `oidctl` startup command, `sslauth` and `wurl`. The `sslauth` is for SSL authentication mode(supports value 1/2/3 for no-auth/one-way or two-way authentication) and `wurl` is the location of wallet to connect to consumer node.

Sample `oidctl` command to bring up replication server in two-way authentication connectivity to consumer node is:

```
oidctl connect=inst2 server=oidrepld inst=1 flags=" -h localhost -p 3132 -sslauth
3 -wurl<location of auto-login wallet> start”
```

41.1.10 LDAP Replication Filtering for Partial Replication

This section describes rules and best practices to follow when specifying naming contexts in LDAP partial replication. It contains the following topics:

- Filtering of Naming Contexts in LDAP Replication
- Attributes that Control Naming Contexts
- Filtering Rules for Naming Contexts in LDAP Replication
- Scenarios of Filtering Naming Context in LDAP Replication
- Rules for Including or Excluding Naming Contexts and Attributes
- Optimization of Partial Replication Naming Context for Better Performance

> **See Also:**
>
> How to Decide Full or Partial Content Replication?.

41.1.10.1 Filtering of Naming Contexts in LDAP Replication

In LDAP-based replication, you can include a given naming context for replication and exclude one or more of the subtrees within that naming context from replication. You can also exclude from replication one or more of the attributes in that naming context.

In LDAP-based replication, only naming contexts explicitly specified as included are replicated.

41.1.10.2 Attributes that Control Naming Contexts

The attributes that control naming contexts are described in Understanding Replication Naming Context Object Entry.

41.1.10.3 Filtering Rules for Naming Contexts in LDAP Replication

When two or more naming context objects are configured for replication, the filtering rules are as follows:

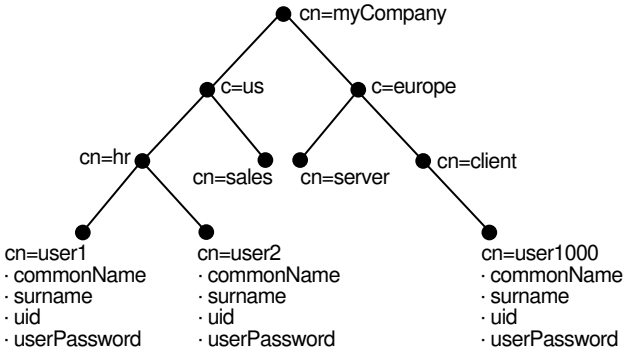
- The overall included naming context is the union of all included naming contexts defined in each naming context object.
- The overall excluded naming contexts is the union of all excluded naming contexts defined in each naming context object.

- The attribute exclusions in a naming context object are specific only to that naming context object.
- If there is a conflict between an included naming context and an excluded naming context, the excluded naming context overrules the included naming context. For example, if an included naming context in naming context object A is a subtree of an excluded naming context specified in another naming context object, B, the subtrees specified in `orclexcludednamingcontexts` of naming context object B are not replicated. That is, replication filtering in naming context object A is ignored.
- If you configure partial replication between two different versions of Oracle Internet Directory (for example 10g (10.1.4.0.1) and 11g Release 1 (11.1.1.0.0)), then you cannot exclude a naming context. Instead, you must explicitly specify the naming contexts to be replicated as included naming contexts.

41.1.10.4 Scenarios of Filtering Naming Context in LDAP Replication

The discussion in this section relies on the sample naming context illustrated in Figure 41-1. A partial list of user attributes is shown under `cn=user1`, `cn=user2`, and `cn=user1000`.

Figure 41-1 A Sample Naming Context



> **See Also:**
>
> Oracle Directory Replication Schema Elements in *Reference for Oracle Identity Management* for descriptions of the attributes in the replication naming context entry

The following examples show how these rules work:

- Scenario A: Included Naming Context is a Subtree of an Included Naming Context in Another Object
- Scenario B: Included Naming Context is a Subtree of an Excluded Naming Context in Another Object

41.1.10.4.1 Scenario A: Included Naming Context is a Subtree of an Included Naming Context in Another Object

Scenario A: The Included Naming Context in One Naming Context Object Is a Subtree of the Included Naming Context in Another Naming Context Object

In this scenario, the included naming context in naming context object #2 is a subtree of the included naming context in object #1.

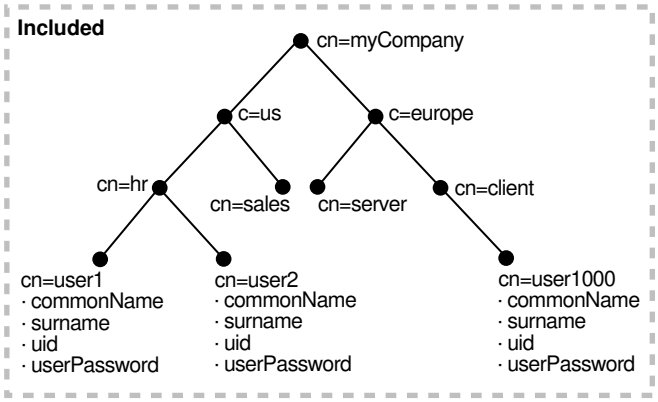
Naming Context Object #1

```
dn:cn=namectx001,
 cn=replication namecontext,
 orclagreementid=unique_identifier_of_the_replication_agreement,
 orclreplicaid=unique_identifier_of_the_supplier,
 cn=replication configuration

orclincludednamingcontexts: cn=mycompany
```

Naming context object #1 includes the entire DIT under `cn=myCompany`, as shown in Figure 41-2.

Figure 41-2 Naming Context Object #1



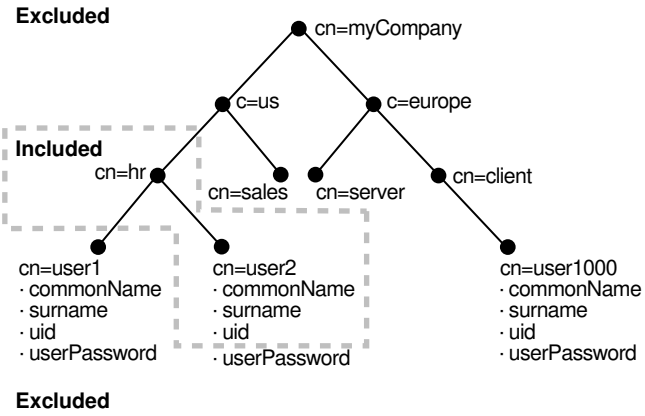
Naming Context Object #2

```
dn:cn=namectx002,
 cn=replication namecontext,
 orclagreementid=unique_identifier_of_the_replication_agreement,
 orclreplicaid=unique_identifier_of_the_supplier,
 cn=replication configuration

orclincludednamingcontexts: cn=hr,c=us,cn=mycompany
orclexcludednamingcontexts: cn=user1,cn=hr,c=us,cn=mycompany
orclexcludedattributes: userPassword
```

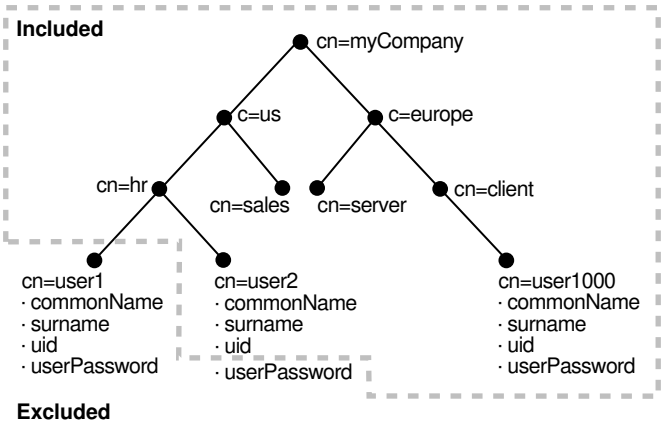
Naming context object #2 includes the DIT under `cn=hr,c=us,cn=mycompany`, but excludes `cn=user1` and the attribute `userPassword`, as shown in Figure 41-3.

Figure 41-3 Naming Context Object #2



The result of combining naming context objects #1 and #2 is shown in Figure 41-4.

Figure 41-4 Result of Combining Naming Context Objects #1 and #2



In this scenario, the naming context that is replicated is the highest one specified in the `orclincludednamingcontexts` attribute. Any excluded naming contexts are not replicated. All changes under the subtree `cn=mycompany` are replicated, except for `cn=user1,cn=hr,c=us,cn=mycompany` and the attribute `userPassword` under `cn=hr,c=us,cn=mycompany`, which are excluded. The attribute `userPassword` under the rest of the DIT, however, is not excluded from replication because exclusion of userPassword was specified only for naming context object #2, which only included the DIT under `cn=hr`.

41.1.10.4.2 Scenario B: Included Naming Context is a Subtree of an Excluded Naming Context in Another Object

Scenario B: The Included Naming Context in One Naming Context Object Is a Subtree of An Excluded Naming Context in Another Naming Context Object

In this scenario, the excluded naming context in naming context object #4 is a subtree of the excluded naming context defined in naming context object #3.

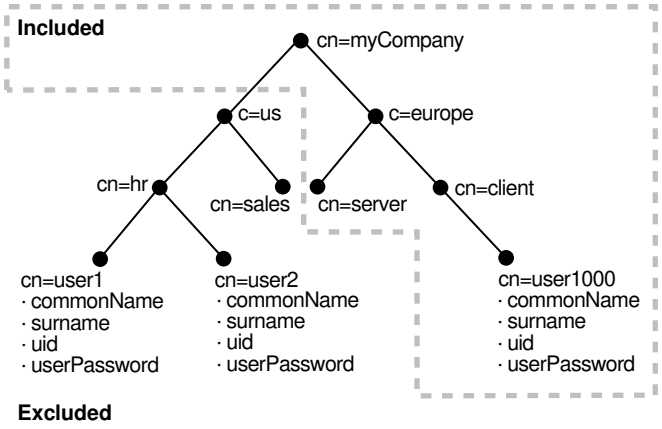
Naming Context Object #3

```
dn:cn=namectx001,cn=replication namecontext,
 orclagreementid=identifier,orclreplicaid=supplier,cn=replication configuration

orclincludednamingcontexts: cn=mycompany
orclexcludednamingcontexts: c=us,cn=mycompany
```

Naming context object #3 excludes everything under `c=us,cn=mycompany`, as shown in Figure 41-5.

Figure 41-5 Naming Context Object #3



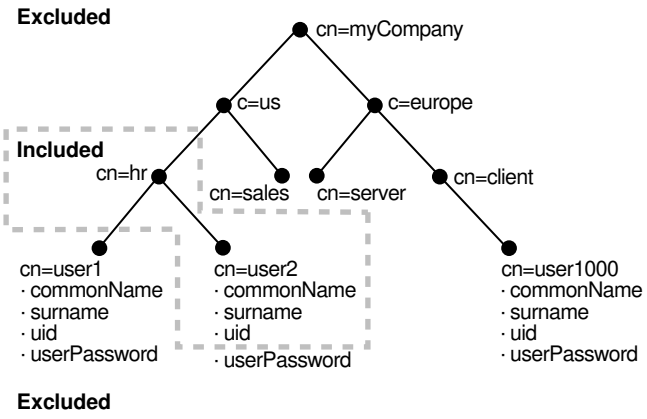
Naming Context Object #4

```
dn:cn=namectx002,cn=replication
 namecontext,orclagreementid=identifier,orclreplicaid=supplier,
 cn=replication configuration

orclincludednamingcontexts: cn=hr, c=us,cn=mycompany
orclexcludednamingcontexts: cn=user1,cn=hr,c=us,cn=mycompany
orclexcludedattributes: userPassword
```

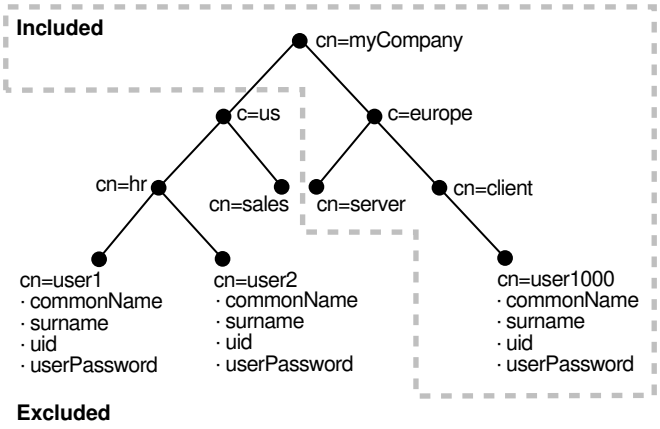
Naming context object #4 includes the DIT under `cn=hr,c=us,cn=mycompany` but excludes `user1`, and the `userPassword` attribute for all users, as shown in Figure 41-6.

Figure 41-6 Naming Context Object #4



The result of combining naming context objects #3 and #4 is shown in Figure 41-7.

Figure 41-7 Result of Combining Naming Context Objects #3 and #4



In this scenario, the included naming context specified in naming context object #4 is not replicated. That naming context is a subtree of a specified excluded naming context in naming context object #3. In this case, naming context object #4 is ignored, and no changes under `cn=hr,c=us,cn=mycompany` are replicated.

41.1.10.5 Rules for Including or Excluding Naming Contexts and Attributes

Oracle Internet Directory has rules to include and exclude naming contexts and attributes.

The following naming contexts cannot be replicated:

- DSE root-specific entry
- `orclagreementid=000001,cn=replication configuration`
- `cn=subconfigsubentry`
- `cn=Oracle Internet Directory`
- `cn=subregistrysubentry`

The following naming contexts cannot be excluded from replication:

- `cn=catalogs`
- `cn=subschemasubentry`
- `cn=oracleschemaversion`
- `cn=replication configuration`

The following attributes cannot be excluded from replication whether they are mandatory or optional. Even if you specify attributes in this list for exclusion from replication, they are always replicated.

- `orclguid`
- `creatorsname`
- `createtimestamp`
- `cn`

- dn
- attributetypes
- objectclasses
- objectclass
- orclindexedattribute
- orclproductversion

You cannot exclude mandatory attributes from replication. For example, suppose that you have an object class named my_object_class, which includes the following attributes: mandatory_attribute_1, optional_attribute_1, and optional_attribute_2. In this case, you cannot exclude from replication mandatory_attribute_1.

If you attempt to exclude from replication an attribute that is a mandatory attribute for an entry, replication server still replicates that attribute.

In partial replication, when a naming context is changed from included to excluded using the moddn operation, the replication server deletes the naming context at the consumer. Similarly, if the naming context is changed from excluded to included by using modn at the supplier, then the replication server synchronizes the entire naming context from supplier to consumer.

41.1.10.6 Optimization of Partial Replication Naming Context for Better Performance

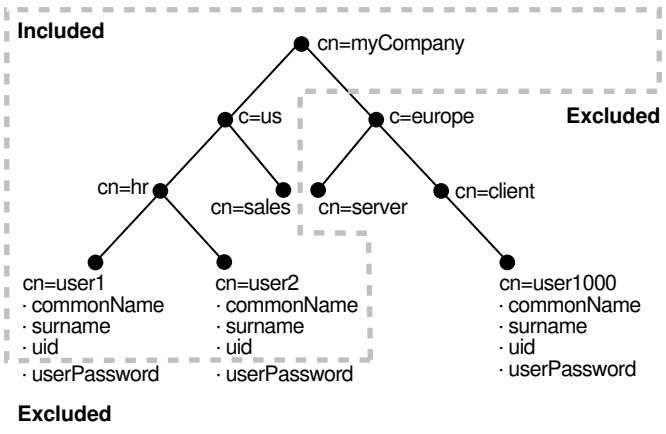
You must plan partial replication carefully to avoid degrading the performance of the replication process. For best performance, use as few naming context objects as possible. For example, the combined use of naming context objects #5 and #6 fulfills the same requirement as the use of naming context object #7, but using naming context object #7 provides better performance.

Naming Context Object #5

```
cn=namectx001,cn=replication
namecontext,orclagreementid=identifier,orclreplicaid=supplier,cn=replication
configuration
orclincludednamingcontexts: cn=mycompany
orclexcludednamingcontexts: c=europe,cn=mycompany
orclexcludedattributes: userPassword
```

Naming context object #5 is shown it Figure 41-8. It includes the DIT under cn=mycompany, but excludes everything under c=europe. It also excludes the attribute userPassword.

Figure 41-8 Naming Context Object #5

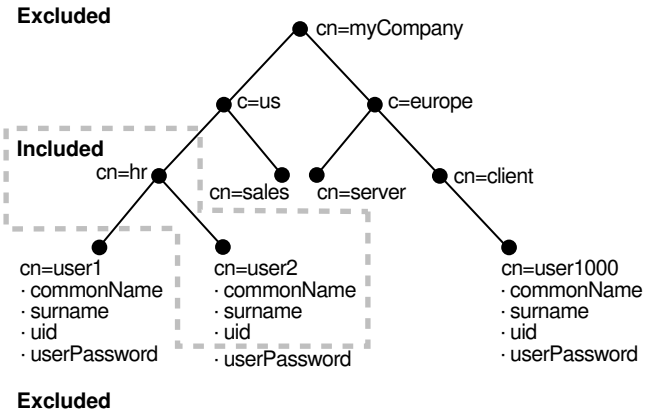


Naming Context Object #6

```
cn=namectx002,cn=replication
namecontext,orclagreementid=<id>,orclreplicaid=<supplier>,cn=replication configuration
orclincludednamingcontexts: cn=hr, c=us,cn=mycompany
orclexcludednamingcontexts: cn=user1,cn=hr, c=us,cn=mycompany
orclexcludedattributes: userPassword
```

Naming context object #6 is shown in Figure 41-9. It includes the DIT under `cn=hr, c=us, cn=mycompany` but excludes `user1` and the attribute `userPassword`.

Figure 41-9 Naming Context Object #6



If naming context objects #5 and #6 are combined, then all changes under `cn=mycompany` are replicated, except for `c=europe,c=mycompany`, `cn=user1,cn=hr,c=us,cn=mycompany`, and the attribute `userPassword`.

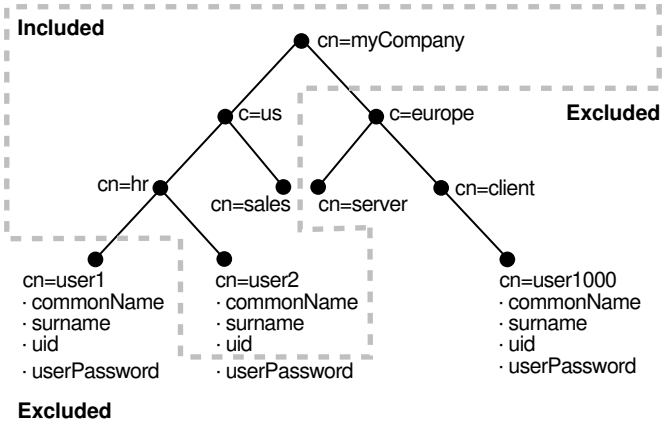
You could fulfill the same requirement, however, by using naming context object #7. Using a single naming context object provides better partial replication performance.

Naming Context Object #7

```
cn=namectx001,cn=replication
namecontext,orclagreementid=identifier,orclreplicaid=supplier,cn=replication
configuration
orclincludednamingcontexts: cn=mycompany
orclexcludednamingcontexts: c=europe,cn=mycompany
orclexcludednamingcontexts: cn=user1,cn=hr, c=us,cn=mycompany
orclexcludedattributes: userPassword
```

Naming context object #7 is shown in Figure 41-10.

Figure 41-10 Naming Context Object #7



41.2 Testing Replication by Using Oracle Directory Services Manager

Use Oracle Directory Services Manager to test directory replication by doing the following:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Data Browser**.
3. Create a single entry on the MDS node, as described in Adding a New Entry by Using Oracle Directory Services Manager.

   The identical entry appears in approximately 1 to 10 minutes on the RMS. You can adjust the timing in the replication server configuration set entry. If entries are modified on any nodes in the DRG, then the changes are replicated.

41.3 Setting Up a LDAP-Based Replication by Using the Command Line

You can setup a LDAP-based replication by using the command line.

This section contains these topics:

- Copying Your LDAP Data by Using ldifwrite and bulkload
- Setting Up an LDAP-Based Replica with Customized Settings
- Deleting an LDAP-Based Replica

> **See Also:**
>
> Replication Direction: One-Way, Two-Way, or Peer to Peer

41.3.1 Copying Your LDAP Data by Using ldifwrite and bulkload

You can use `ldifwrite` and `bulkload` to copy LDAP data from one host to another.

Use the `ldifwrite` utility to back up LDAP data with operational attributes preserved. After this is done, use the `bulkload` utility to load data to all replicas in a group.

Use `bulkload` with the `check="TRUE"`, `generate="TRUE"`, and `restore="TRUE"` arguments, and then with the `load="TRUE"` argument. Preserve the operational attributes by using the same intermediate files (generated by using the `generate="TRUE"` argument) for all replicas. You can load multiple replicas in the same invocation of the `bulkload` command by using `connect="`*`connect_string`*`"` with the appropriate connect string for each replica.

Using this method can take a long time for a directory with one million entries.

> **See Also:**
>
> - Performing Bulk Operations
> - Backing Up and Restoring Oracle Internet Directory
> - Command-Line Tools Overview in *Reference for Oracle Identity Management*

41.3.2 Setting Up an LDAP-Based Replica with Customized Settings

To establish customized settings, you must first install the new node.

To install the new node, follow the instructions in *Installing and Configuring Oracle Internet Directory*.

After configuring LDAP-based replication with `remtool`, you can customize the `namingcontext` defining what is replicated for that LDAP-based node.

> **See Also:**
>
> The discussion of naming contexts in Implications of LDAP-Based Partial Replication.

There are two ways to set up an LDAP-based replica with customized setting, based on how you will migrate the data from the directory:

- Use the command-line tools. Use `ldifwrite` to backup the data from the supplier replica, then use `bulkload` to restore the data to the consumer replica
- Use automatic bootstrapping. This is a replication server feature that automatically bootstrap the data from the supplier replica to the consumer replica, based upon replication configuration.

Table 41-1 compares these two methods.

The following sections explain these methods further:

- Data Migration Using ldifwrite/bulkload versus Automatic Bootstrapping

- Setting Up an LDAP-Based Replica by Using Automatic Bootstrapping
- Setting Up an LDAP-Based Replica by Using the ldifwrite Tool

41.3.2.1 Data Migration Using ldifwrite/bulkload versus Automatic Bootstrapping

You must use the ldifwrite/bulkload or automatic bootstrapping methods for data migration for different fitting scenarios.

Table 41-1 Data Migration Using ldifwrite/bulkload versus Automatic Bootstrapping

Migration Using ldifwrite/bulkload	Migration Using Automatic Bootstrapping
Manual procedure	Automatic procedure
Faster performance	Uses the filtering capability of partial replication
Good for a large amount of data	Good for a smaller number of entries

> **Note:**
>
> You must use Command-line Tools to Setup and Modify Replication (`ldifwrite` and `bulkload`) for the following scenario:
>
> - Replication for a directory that has more than 100,000 entries
>
> For other scenarios, you can use either migration method.

41.3.2.2 Setting Up an LDAP-Based Replica by Using Automatic Bootstrapping

The following eight tasks enable you to configure an LDAP-based replica by using automatic bootstrapping. They are explained in the paragraphs that follow this list.

- Identifying and Starting the Directory Server on the Supplier Node
- Creating the New Consumer Node by Installing Oracle Internet Directory
- Backing Up Metadata from the New Consumer Node
- Adding a LDAP-Based Replica by Using the Replication Environment Management Tool
- Configuring the Consumer Replica for Automatic Bootstrapping
- Changing Default Replication Parameters
- Ensuring the Directory Replication Servers are Started

41.3.2.2.1 Identifying and Starting the Directory Server on the Supplier Node

You can identify the supplier for an LDAP-based replica. The supplier can be

- A directory that is not a member of any replication group
- A node of a multimaster replication group
- Another LDAP-based replica

Make sure the Oracle Internet Directory server is started on the Supplier node. To start the directory server, type the following command:

```
start(name='instance-name',type='OID')
```

41.3.2.2.2 Creating the New Consumer Node by Installing Oracle Internet Directory

Install a new Oracle Internet Directory on the replica, as documented in *Installing and Configuring Oracle Internet Directory*.

41.3.2.2.3 Backing Up Metadata from the New Consumer Node

Before configuring the new node as an LDAP-based replica with customized settings, you must first migrate its metadata to the supplier node, as follows:

- Make sure the Oracle Internet Directory server is up and running on both the supplier node and the new node created, as discussed in Creating the New Consumer Node by Installing Oracle Internet Directory, so that the backup process (remtool –backupmetadata) can succeed.
- From the newly created node, run the following command:

```
remtool -backupmetadata \
        -replica "new_node_host:new_node_port" \
        -master "master_host:master_port"
```

  where *master_host*:*master_port* are the hostname and port number for the desired replica's supplier. you are prompted for replication DN password.

> **See Also:**
>
> The `remtool`command-line tool reference in *Reference for Oracle Identity Management* for more information about `remtool` options, including `-backupmetadata`.

- Apart from loading the metadata into master replica, this command creates a file named `ocbkup.`*`new_replica_id`*`.TO.`*`master_replicaid`*`.`*`timestamp`*`.ldif` containing the metadata as back up. This file is created under the *`$DOMAIN_HOME`*`/tools/OID/logs` directory. This file contains the changes made to master replica in LDIF format, a copy of the SSO container entry [orclApplicationCommonName=ORASSO_SSOSERVER, cn=SSO, cn=Products, cn=OracleContext] and DAS URL container entry [cn=OperationURLs, cn=DAS, cn=Products, cn=OracleContext].
- If the metadata backup succeeds, it shows the following message in the terminal:

```
Backup of metadata will be stored in $DOMAIN_HOME/tools/OID/logs/
ocbkup.replicaid_pilot.TO.replcicaid_master.timestamp.ldif.

Metadata copied successfully.
```

  The message contains the path of your *`DOMAIN_HOME`* and filename.
- If the metadata backup is unsuccessful, the *$DOMAIN_HOME*`/tools/OID/logs/remtool.log` file contains error messages. If you invoked `remtool` from a terminal, error messages appear on that terminal.

41.3.2.2.4 Adding a LDAP-Based Replica by Using the Replication Environment Management Tool

To add a LDAP-based replica, enter the following on the consumer replica:

```
remtool -paddnode [-v] [-bind supplier_host_name:port]
```

you are prompted for the *replication_dn_password*.

The `remtool` utility prompts for agreement type. Select One-Way, Two-Way, or Multimaster LDAP, depending on which type of replica you are adding.

you are prompted for the replica ID of the supplier. This is the value of the attribute `orclreplicaid` in the root DSE entry. To find the value to supply, type:

```
ldapsearch -D cn=orcladmin -q  -p port -b "" -s base "objectclass=*" orclreplicaid
```

you are prompted for the list of available naming contexts in the supplier replica. If you are planning to set up partial replication between server instances running different versions of Oracle Internet Directory, enter `e`.

After `remtool` has completed successfully, add a separate naming context for each subtree to be replicated, as follows:

1. Determine the replica ID of the supplier and consumer replica nodes by executing the following search command on both nodes:

   ```
   ldapsearch -h host -p port -D cn=orcladmin -q -b "" \
      -s base "objectclass=*" orclreplicaid
   ```

2. Determine the replication agreement entry, which is of the form:

   ```
   "orclagreementid=unique_identifier_of_the_replication_agreeement,orclreplicaid=supplier_replica_id,cn=replication configuration"
   ```

   To find it, type:

   ```
   ldapsearch -D cn=orcladmin -q -h supplier_host -p supplier_port \
      -b "orclreplicaid=supplier_replica_id,cn=replication configuration" \
      -s sub "(&(orclreplicadn=*consumer_replica_id*)
   (objectclass=orclreplagreemententry))" dn
   ```

3. For each naming context that you plan to include in replication, add an entry like the following on both the consumer and supplier replica nodes. The following LDIF file specifies a naming context (subtree) to be included in replication and a some attributes to exclude from replication.

   ```
   dn: cn=includednamingcontext000002,
    cn=replication namecontext,orclagreementid=000003,
    orclreplicaid=stajv18_oid10143,cn=replication configuration
   objectclass: top
   objectclass: orclreplnamectxconfig
   orclincludednamingcontexts: cn=users,dc=small,dc=com
   orclexcludedattributes: userpassword
   orclexcludedattributes: telephonenumber
   cn: includednamingcontext000002
   ```

   Add the LDIF file to Oracle Internet Directory on the consumer and supplier replica using the following LDAP add command:

   ```
   ldapadd -D cn=orcladmin -q -h host -p port -v -f ldif_file
   ```

4. Repeat the previous step for each naming context (subtree) to be configured in partial replication.

> **See Also:**
>
> - The `remtool`command-line tool reference in *Reference for Oracle Identity Management* for more information about the Replication Environment Management Tool
> - LDAP Replication Filtering for Partial Replication

41.3.2.2.5 Configuring the Consumer Replica for Automatic Bootstrapping

To use the automatic bootstrap capability, on the consumer, set the `orclReplicaState` attribute of the consumer replica subentry to `0` as follows:

1. Edit the sample file `mod.ldif` as follows:

```
Dn: orclreplicaid=unique_replicaID_of_consumer, cn=replication configuration
Changetype:modify
replace:orclReplicaState
OrclReplicaState: 0
```

> **Note:**
>
> On Windows systems, ensure that the replication server is not running before you enable bootstrapping by changing the value of `orclReplicaState` to `0`.

2. Use `ldapmodify` at the consumer to update the consumer replica's subentry `orclreplicastate` attribute.

```
ldapmodify -D "cn=orcladmin" -q -h consumer_host -p port -f mod.ldif
```

> **See Also:**
>
> Managing and Monitoring Replication for more information about the bootstrap capability of the LDAP-based replication

41.3.2.2.6 Changing Default Replication Parameters

You can change the default parameters for replication agreements and for the replica subentry.

> **See Also:**
>
> - Managing Replication Configuration Attributes
> - Implications of LDAP-Based Partial Replication

41.3.2.2.7 Ensuring the Directory Replication Servers are Started

The exact procedure for starting the replication servers depends on whether this is a one-way or a two-way replica or multimaster replica.

- For one-way LDAP replication, you must start the replication server at the consumer. For example, to start the replication server at oid1, type:

```
oidctl connect=connStr server=oidrepld instance=1 \
  name=asinst_1 component name=oid1 \
  flags="-h consumer_host -p consumer_port" start
```

> **Note:**
>
> If you are deploying a single master with read-only replica consumers, you can reduce performance overhead by turning off conflict resolution. To do so, change the value of `orclconflresolution` to 0 by using the following ldif file with `ldapmodify`:
>
> ```
> dn: cn=configset0,cn=osdrepld,cn=subconfigsubentry
> changetype: modify
> replace: orclconflresolution
> orclconflresolution: 0
> ```

- For two-way or multimaster LDAP replication, you must start the Oracle Internet Directory replication server at each node, as follows:

```
oidctl connect=connStr server=oidrepld instance=1 \
   name=instance_name componentname=component_name flags="-h \
   LdapHost -p LdapPort" start
```

When the replication server is started, it starts to bootstrap the data from the supplier to the consumer. After the bootstrap has completed successfully, the replication server automatically change to ONLINE mode (`orclreplicastate=1`) to process changes from the supplier to the consumer. You can monitor the value of `orclreplicastate` by using the following command line:

```
ldapsearch -p port-h host -D cn=orcladmin -q \
  -b "orclreplicaid=unique_replicaID_of_consumer, cn=replication configuration" \
  -s base "objectclass=*" orclreplicastate
```

41.3.2.3 Setting Up an LDAP-Based Replica by Using the ldifwrite Tool

This section discuss the general tasks you perform when configuring an LDAP-based replica by using the ldifwrite tool. It contains these topics:

- Starting the Directory Server on Both the Supplier and the Consumer Nodes
- Backing Up Metadata from the New Consumer Node
- Changing the Directory Server at the Supplier to Read-Only Mode
- Adding a LDAP-Based Replica by Using the Replication Environment Management Tool
- Backing Up the Naming Contexts to Be Replicated
- Changing the Directory Server at the Supplier to Read/Write Mode
- Loading the Data on the New Consumer

- Changing Default Replication Parameters
- Ensuring the Directory Replication Servers are Started

41.3.2.3.1 Starting the Directory Server on Both the Supplier and the Consumer Nodes

To start the directory server on both the supplier and the consumer nodes:

1. Identify the supplier for an LDAP-based replica. The supplier can be:
   - A directory that is not a member of any replication group
   - A node of a multi-master replication group
   - Another LDAP-based replica

   Make sure the Oracle Internet Directory server is started on the Supplier node. To start the directory server, type the following command:

   ```
   $DOMAIN_HOME/bin/startComponent.sh <instance-name>
   ```

2. Identify the consumer node, which must be a new Oracle Internet Directory install. To install a new Oracle Internet Directory as a Master, follow the directions in *Installing and Configuring Oracle Internet Directory* Make sure the Oracle Internet Directory server is started on the new consumer node. To start the directory server, type the following command:

   ```
   $DOMAIN_HOME/bin/startComponent.sh <instance-name>
   ```

41.3.2.3.2 Backing Up Metadata from the New Consumer Node

Before configuring the consumer as an LDAP-based replica with customized settings, you must first migrate its metadata to the supplier node, as follows:

- Make sure the Oracle Internet Directory server is up and running on both the supplier node and the new node created, as discussed in Creating the New Consumer Node by Installing Oracle Internet Directory, so that the backup process (remtool –backupmetadata) can succeed.
- From the consumer node, run the following command:

  ```
  remtool -backupmetadata \
          -replica "consumer_host:consumer_port" \
          -master "supplier_host:supplier_port"
  ```

  you are prompted for the passwords.
- Apart from loading the metadata into master replica, this tool creates a file named `ocbkup.consumer_replica_id.TO.supplier_replica_id.timestamp.dat` containing the metadata as back up. This file is created in the *$DOMAIN_HOME*`/tools/OID/logs` directory. This file contains the changes made to the master replica in LDIF format, a copy of SSO container entry [orclApplicationCommonName=ORASSO_SSOSERVER, cn=SSO, cn=Products, cn=OracleContext] and DAS URL container entry [cn=OperationURLs, cn=DAS, cn=Products, cn=OracleContext].
- If the metadata backup succeeded, `remtool` displays the following message in the terminal:

  ```
  Backup of metadata will be stored in $DOMAIN_HOME/tools/OID/logs/
  ocbkup.replicaid_pilot.TO.replcicaid_master.timestamp.ldif.

  Metadata copied successfully.
  ```

  The message contains the path of your *DOMAIN_HOME*.

The message contains the actual path of your *DOMAIN_HOME* and filename.

If the metadata backup is unsuccessful, the *$DOMAIN_HOME*`/tools/OID/logs/remtool.log` file contains error messages. If you invoked `remtool` from a terminal, error messages appear on that terminal.

41.3.2.3.3 Changing the Directory Server at the Supplier to Read-Only Mode

To ensure data consistency, change the directory server on the supplier node to read-only. To switch the server from read/write to read-only mode, use one of the procedures in Changing the Server Mode.

41.3.2.3.4 Adding a LDAP-Based Replica by Using the Replication Environment Management Tool

To add a replica, enter the following on the consumer replica:

```
remtool -paddnode [-v] [-bind supplier_host_name:port]
```

you are prompted for the *replication_dn_password*. The remtool utility prompts for agreement type. Select One-Way or Two-Way LDAP, depending on which type of replica you are adding.

> **See Also:**
>
> The `remtool` command-line tool reference in *Reference for Oracle Identity Management* for more information about the Replication Environment Management Tool

41.3.2.3.5 Backing Up the Naming Contexts to Be Replicated

If there is a large number of entries in the naming contexts that you want to replicate to the LDAP-based replica, then Oracle recommends that you back up these naming contexts at the supplier node and then load them to the LDAP-based replica.

To back up the naming contexts:

1. Identify the replication agreement DN created in "Adding a LDAP-Based Replica by Using the Replication Environment Management Tool".

   ```
   ldapsearch -h supplier_host -p port \
              -b "orclreplicaid=supplier_replicaID,cn=replication configuration" \
              -s sub "(orclreplicadn= orclreplicaid=consumer_replica_ID, \
                       cn=replication configuration)" dn
   ```

2. On the supplier, ensure that `DOMAIN_HOME` is set, the use the following command to get the data from the supplier. Data loaded into the file will be based on the agreement configured:

   ```
   ldifwrite connect="connect_string_of_sponsor_node" \
             baseDN="replication_agreement_dn_retrieved_in_step_1" \
             file="name_of_output_LDIF_file"
   ```

> **See Also:**
>
> - Implications of LDAP-Based Partial Replication
> - The `ldifwrite`command-line tool reference in *Reference for Oracle Identity Management* for more instructions on using ldifwrite to back up part of the naming context

41.3.2.3.6 Changing the Directory Server at the Supplier to Read/Write Mode

If you performed "Changing the Directory Server at the Supplier to Read-Only Mode", then change the directory server on the supplier back to read/write mode. Use one of the procedures in Changing the Server Mode.

41.3.2.3.7 Loading the Data on the New Consumer

To load data on the new consumer:

1. If there are multiple files, then combine them into one file—for example, `backup_data.ldif`.
2. If naming contexts exist on the LDAP-based consumer replica, then remove them by using bulkdelete. Ensure `DOMAIN_HOME` is set, then enter the following:

   ```
   bulkdelete connect="connect_string_of_replica" baseDN="naming_context"
   ```

Perform this step for each naming context that was backed up in "Backing Up the Naming Contexts to Be Replicated".

On the consumer, load the data to the replica by using bulkload in the append mode. Ensure `DOMAIN_HOME` is set, then enter the following:

```
bulkload connect="connect_string_of_replica" append="TRUE" check="TRUE" \
   generate="TRUE" restore="TRUE" file="backup_data.ldif"

bulkload connect="connect_string_of_replica" load="TRUE"
```

> **See Also:**
>
> - The `bulkload` command-line tool reference in *Reference for Oracle Identity Management* for instructions on using bulkload in either the default mode or the append mode
> - The `bulkdelete` command-line tool reference in *Reference for Oracle Identity Management*

41.3.2.3.8 Changing Default Replication Parameters

You can change the default parameters for replication agreements, for the replica subentry, and for the replication naming context configuration objects. This task is optional.

> **See Also:**
>
> - Managing Replication Configuration Attributes
> - Implications of LDAP-Based Partial Replication

41.3.2.3.9 Ensuring the Directory Replication Servers are Started

The exact procedure for starting the replication servers depends on whether this is a one-way or a two-way replica.

- For one-way LDAP replication, you must start the replication server at the consumer. Type:

```
oidctl connect=connStr server=oidrepld instance=1 \
   name=instance_name componentname=oidComponentName \
   flags="-h LdapHost -p LdapPort" start
```

> **Note:**
>
> If you are deploying a single master with read-only replica consumers, you can reduce performance overhead by turning off conflict resolution. To do so, change the value of `orclconflresolution` to 0 by using the following ldif file with `ldapmodify`:
>
> ```
> dn: cn=configset0,cn=osdrepld,cn=subconfigsubentry
> changetype: modify
> replace: orclconflresolution
> orclconflresolution: 0
> ```

- For two-way LDAP replication, you must start the Oracle Internet Directory replication servers at both the sponsor replica and the new replica, as follows:

  1. Start or restart the replication server at the sponsor replica. Type:

     ```
     oidctl connect=connStr server=oidrepld instance=1 \
        name=instance_name componentname=component_name \
        flags="-h LdapHost -p LdapPort" start
     ```

  2. Start the replication server at the new replica. Type:

     ```
     oidctl connect=connStr server=oidrepld instance=1 \
        name=instance_name componentname=oidComponentName \
        flags="-h LdapHost -p LdapPort" start
     ```

41.3.3 Deleting an LDAP-Based Replica

This section explains how to delete an LDAP-based replica.

This section contains the following topics:

- Stopping the Directory Replication Server on the Node to be Deleted
- Deleting the Replica from the Replication Group

> **Note:**
>
> You cannot delete a replica if it is a supplier for another replica. To delete such a replica, you must first delete all its consumers from the replication group.

41.3.3.1 Stopping the Directory Replication Server on the Node to be Deleted

Stop the Oracle directory replication server by typing:

```
oidctl connect=connStr server=oidrepld instance=1 componentname=oidComponentName \
 flags="-h LdapHost -p LdapPort" stop
```

41.3.3.2 Deleting the Replica from the Replication Group

You can delete the replica from the replication group by using the Replication Environment Management Tool. Enter:

```
remtool -pdelnode [-v] [-bind hostname:port_number]
```

You are prompted for the password. Enter a value for *replication_dn_password*.

> **See Also:**
>
> The `remtool`command-line tool reference in *Reference for Oracle Identity Management*

41.4 Scenario: Setting Up a Multimaster Replication Group with Fan-Out

The following topics describe how to set up a multimaster replication group:

- Understanding Multimaster Replication
- Setting Up the Multimaster Replication Group for Node1 and Node2
- Configuring the Replication Agreement
- Starting the Replication Servers on Node1 and Node2
- Testing the Directory Replication Between Node1 and Node2
- Installing and Configuring Node3 as a Partial Replica of Node2
- Customizing the Partial Replication Agreement
- Starting the Replication Servers on All Nodes in the DRG
- Installing and Configuring Node4 as a Full Replica of Node2
- Testing the Replication from Node2 to Node4
- Installing and Configuring Node5 as a Two-Way Replica of Node1
- Testing the Two-Way Replication Between Node1 and Node5

41.4.1 Understanding Multimaster Replication

To help you set up a multimaster replication group with fan-out, this section offers an example with four systems.

See Table 41-2.

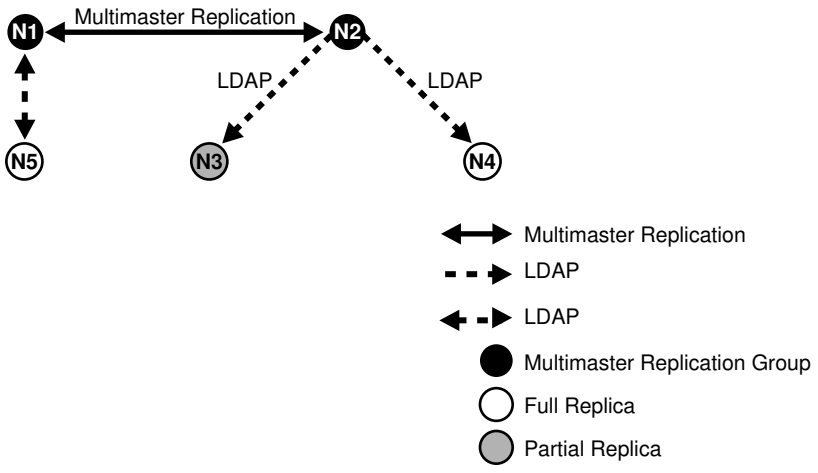
Table 41-2 Nodes in Example of Partial Replication Deployment

Node	Host Name	Port
Node1	mycompany1.com	3000
Node2	mycompany2.com	4000
Node3	mycompany3.com	5000
Node4	mycompany4.com	6000
Node5	mycompany5.com	7000

In this example, the user has set up the following requirements:

- **Requirement 1:** Node1 and Node2 must be synchronized so that changes made on either node are replicated to the other— but the naming context `cn=private users, cn=mycompany` is to be excluded from this replication.
- **Requirement 2:** The naming context `ou=Americas,cn=mycompany` on node3 is to be partially synchronized from Node2 so that only changes made under `ou=Americas, cn=mycompany` on Node2 are replicated to Node3. The following are to be excluded from this replication:
  - Changes made under `cn=customer profile, ou=Americas, cn=mycompany`
  - Changes in the attribute `userpassword`.
- **Requirement 3:** Node4 is to be configured as a full replica of node 2, that is, changes to all naming contexts in Node2 are replicated (one-way) to Node4.
- **Requirement 4:** Node5 is to be configured as a two-way (updatable) full replica of Node1.

Figure 41-11 Example of Fan-Out Replication



To meet the first requirement in this example, we set up a multimaster replication group for Node1 and Node2. To meet the second, we set up a partial replica for Node2 and Node3, and for the third, full LDAP replication from Node2 to Node4.

41.4.2 Setting Up the Multimaster Replication Group for Node1 and Node2

This section gives information to set up an LDAP-based multimaster replication group for Node1 and Node2,

To set up an LDAP-based multimaster replication group for Node1 and Node2, follow the instructions in Setting Up an LDAP-Based Replica with Customized Settings

41.4.3 Configuring the Replication Agreement

In the replication agreement between Node1 and Node2, specify the value for the `orclExcludedNamingcontexts` attribute as `cn=private users,cn=mycompany`.

Perform the following steps:

1. Edit the example file `mod.ldif` as follows:

```
dn: orclAgreementID=000001,cn=replication configuration
Changetype:modify
Replace: orclExcludedNamingcontexts
orclExcludedNamingcontexts: cn=private users,cn=mycompany
```

2. Use ldapmodify to update the replication agreement `orclExcludedNamingcontexts` attribute at both Node1 and Node2. To do this, enter:

```
ldapmodify -D "cn=orcladmin" -q -h mycompany1.com -p 3000 -f mod.ldif
ldapmodify -D "cn=orcladmin" -q -h mycompany2.com -p 4000 -f mod.ldif
```

41.4.4 Starting the Replication Servers on Node1 and Node2

To start replication servers on node1 and node2, if you are using LDAP-based replication, follow the instructions on

To start replication servers on node1 and node2, if you are using LDAP-based replication, follow the instructions on

41.4.5 Testing the Directory Replication Between Node1 and Node2

Follow the instructions given in this section.

To do this, follow the instructions in Testing Replication by Using Oracle Directory Services Manager .

41.4.6 Installing and Configuring Node3 as a Partial Replica of Node2

If you want to use the bootstrap capability of partial replication, then follow the instructions given in this section.

If you want to use the bootstrap capability of partial replication, then follow Tasks 1 through 5 in Setting Up an LDAP-Based Replica by Using Automatic Bootstrapping.

If you want to configure the replica by using the ldifwrite tool, then follow Tasks 1 through 9 in Setting Up an LDAP-Based Replica by Using the ldifwrite Tool.

Identify Node2 as the supplier and Node3 as the consumer.

41.4.7 Customizing the Partial Replication Agreement

You can customize the partial replication agreement as described in this section.

- To achieve Requirement 2 in this example, the default replication between Node2 and Node3 must be configured first:

  In partial replication, the `cn=oraclecontext` naming context is replicated by default. You can choose not to replicate it by deleting it at both the supplier (Node2, mycompany2.com) and the consumer (Node3, mycompany3.com).

```
ldapdelete -D "cn=orcladmin" -q -h mycompany2.com \
           -p 4000 "cn=includednamingcontext000001, \
                    cn=replication namecontext,orclagreementid=000002, \
                    orclreplicaid==node2_replica_id, \
                    cn=replication configuration"

ldapdelete -D "cn=orcladmin" -q -h mycompany3.com \
           -p 5000 "cn=includednamingcontext000001, \
                    cn=replication namecontext,orclagreementid=000002, \
                    orclreplicaid==node2_replica_id, \
                    cn=replication configuration"
```

- To replicate the naming context `ou=Americas,cn=mycompany`, and to exclude from replication the naming context `cn=customer profile, ou=Americas, cn=mycompany` and the attribute `userpassword`, create a naming context object as follows:

  1. Edit the example file `mod.ldif` as follows:

```
dn: cn=includednamingcontext000002,cn=replication namecontext,
 orclagreementid=000002,orclreplicaid=node2_replica_id,
 cn=replication configuration
orclincludednamingcontexts: ou=Americas,cn=mycompany
orclexcludednamingcontexts: cn=customer profile, ou=Americas, cn=mycompany
orclexcludedattributes: userpassword
objectclass: top
objectclass: orclreplnamectxconfig
```

  2. Use ldapadd to add the partial replication naming context object at both Node2 and Node3.

```
ldapadd -D "cn=orcladmin" -q -h mycompany2.com -p 4000 -f mod.ldif
ldapadd -D "cn=orcladmin" -q -h mycompany3.com -p 5000 -f mod.ldif
```

- If you decide to use the automatic bootstrap capability of partial replication, then edit the example file `mod.ldif` and use `ldapmodify` to modify the partial replica `orclreplicastate` attribute at both Node2 and Node3, as described in "Configuring the Consumer Replica for Automatic Bootstrapping".

41.4.8 Starting the Replication Servers on All Nodes in the DRG

Start the replication server at each node before starting the replication process.

To do this, start the replication server at each node.

Type:

```
oidctl connect=connStr server=oidrepld instance=1 \
   componentname=oidComponentName flags="-h LdapHost -p LdapPort" start
```

41.4.9 Installing and Configuring Node4 as a Full Replica of Node2

Full replica replication is the default configuration when the new node is installed as an LDAP replica.

Since full replica replication is the default configuration when the new node is installed as an LDAP replica, use the instructions in Setting Up a LDAP-Based Replication by Using the Command Line. When the installer prompts for the supplier information, provide the supplier hostname, `mycompany2.com`, the supplier port, `4000`, and the superuser password.

41.4.10 Testing the Replication from Node2 to Node4

Test this replication using the instructions in the section entitled.

Test this replication using the instructions in the section entitled.

41.4.11 Installing and Configuring Node5 as a Two-Way Replica of Node1

Follow the instructions given in this section.

Follow the instructions in Setting Up a LDAP-Based Replication by Using the Command Line. Provide the supplier hostname, `mycompany1.com`, the supplier port, `3000`, and the superuser password.

41.4.12 Testing the Two-Way Replication Between Node1 and Node5

Follow the instructions given in this section.

Create an entry at Node1 by using the `ldapadd` command-line tool. Wait for the entry to be replicated at Node5. After the entry is replicated to Node5, apply a change to that entry at Node5 using the `ldapmodify` command-line tool. The change is replicated to Node1.

42
Setting Up Replication Failover

This chapter describes replication failover for Oracle Internet Directory, including how to perform a stateless replication failover and a time-based replication failover.
This chapter includes the following sections:

- Introduction to Replication Failover
- Performing a Stateless Replication Failover
- Performing a Time-Based Replication Failover

42.1 Introduction to Replication Failover

The Oracle Internet Directory has supported failover of LDAP replicas from one supplier to another. Administrator intervention is required.

The following sections explain replication failover in detail:

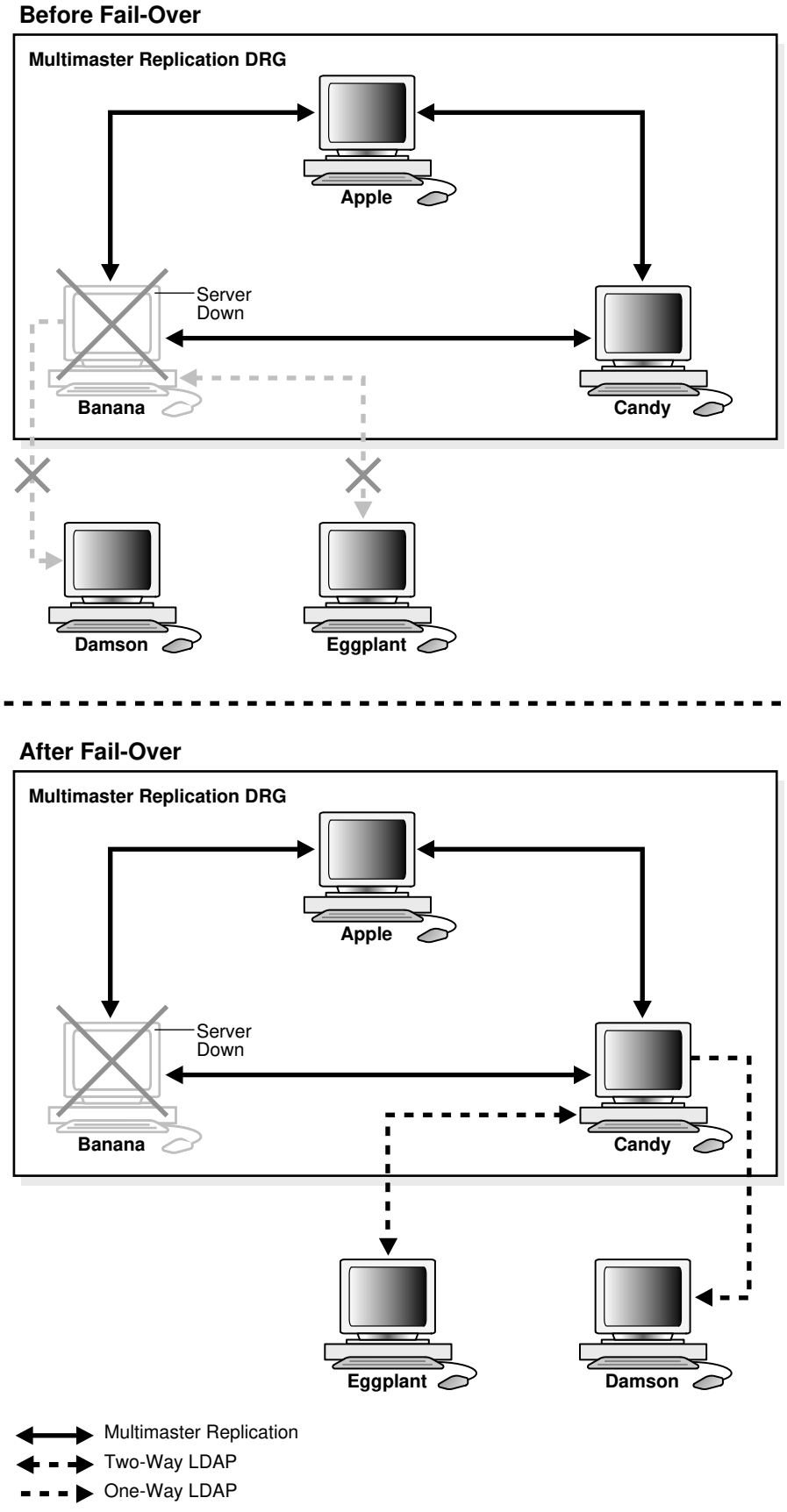
- Replication Failover Scenario
- Supported Failover Topology Types
- Limitations and Warnings for Replication Failover
- Types of Replication Failover

42.1.1 Replication Failover Scenario

This section describes the replication failover scenarios.

Figure 42-1 shows a typical failover scenario.

Figure 42-1 Replication Failover Scenario



This scenario has the following features:

- Apple, Banana and Candy are multimaster replicas in the same DRG.
- Damson is a read-only fan-out replica of Banana. That is, it is a partial replica using one-way LDAP replication.
- Eggplant is an updatable fan-out replica of Banana. That is, it is a partial replica using two-way LDAP replication.
- If Banana goes down, replication between the multimaster DRG and its fan-out replicas is broken.

  An administrator can switch Eggplant and Damson to a new supplier, Candy.

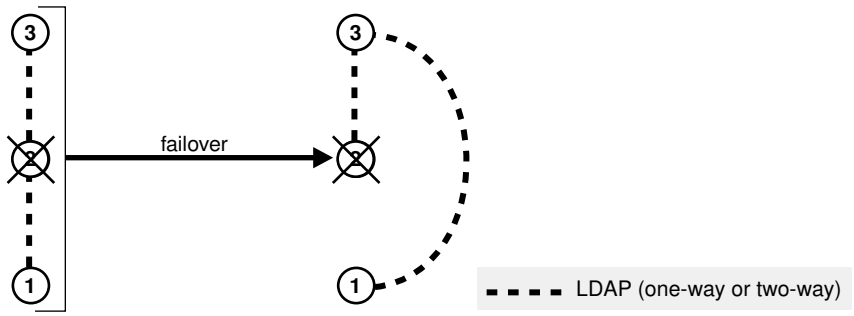
42.1.2 Supported Failover Topology Types

This section describes the supported failover topology types supported by Oracle Internet Directory.

Only two failover topology types are supported:

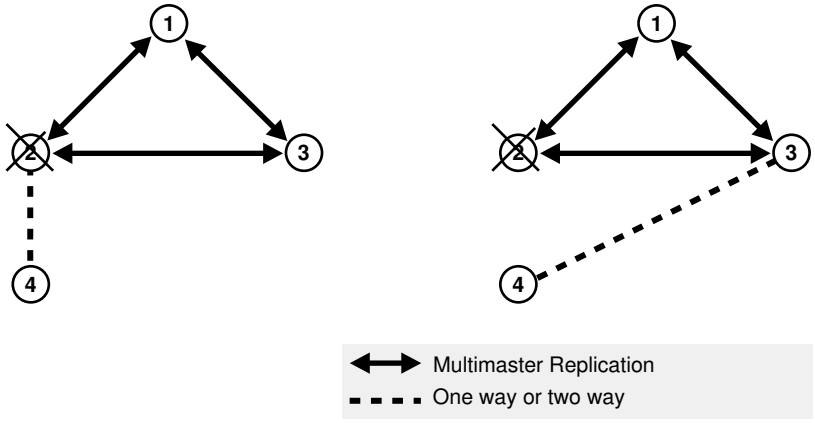
- The consumer and new supplier are both connected to the old supplier with LDAP-based replication agreements. This is shown in Figure 42-2. Node 1 and Node 3 both have LDAP replication agreements with Node 2. Node 2 is the original supplier for Node 1. When Node 2 fails, you can fail over Node 1 to a new supplier, Node 3.

Figure 42-2 Consumer and New Supplier Connected to Old Supplier by LDAP



- The consumer is connected to the old supplier with an LDAP-based agreement and the old supplier is in the same Advanced Replication group as the new supplier. This is shown in Figure 42-3. Node 2 and Node 3 are in the same Advanced Replication DRG. Node 2 is the original supplier for Node 4. When Node 4 fails, you can fail over Node 4 to a new supplier, Node 3.

Figure 42-3 Old and New Suppliers in the Same Directory Replication Group

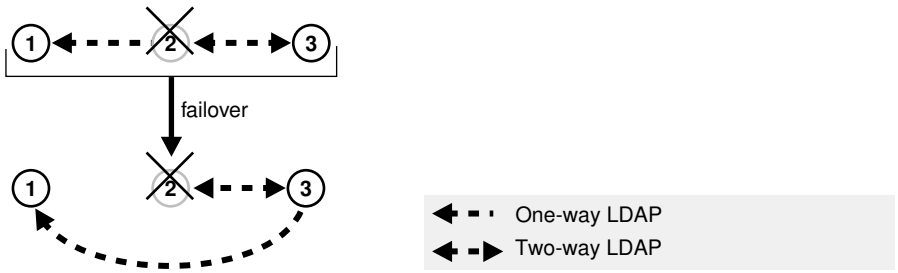


42.1.3 Limitations and Warnings for Replication Failover

This section describes limitations and warnings related to the use of replication failover.

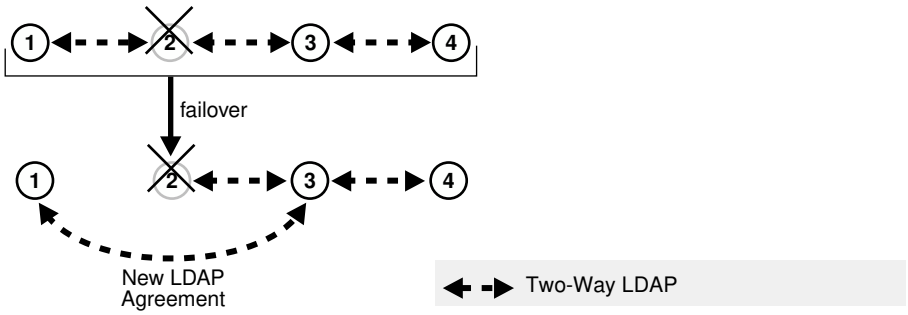
- Replication failover requires administrator intervention.
- Following failover, you must compare and reconcile the consumer with the new supplier.
- The new agreement must be of the same type and direction as the old agreement.
- Only two topology types are supported.
- When a supplier fails, its directly connected replica can only fail over to another directly connected replica of the failed supplier.
- The replication filtering policy for the agreement between the new supplier and old supplier must match that between the old supplier and consumer.
- In most cases, you should fail over the replica in a way that preserves the original replica type. In the case shown in Figure 42-4, node 2 is the old supplier for both node 1 and 3, and node 1 is read-only. When node 2 fails, you could, in theory, set up either node 1 or 3 as the new supplier node. Best practice, however, is to fail over node 1 so that node 3 is the supplier. This preserves node 1's original, read-only replica type.

Figure 42-4 Failover Preserving Replica Type



- If the new agreement is a two-way agreement, after you compare and reconcile the consumer with its new supplier, you must also compare and reconcile all other replicas that are connected to the new supplier with the new supplier. For example, in Figure 42-5, Node 2 has a two-way agreement with Node 3. Node 3 is connected to another replica, Node 4. When Node 2 fails, you set up a two-way agreement between node 3 and node 1. After comparing and reconciling node 3 with node 1, you must also compare and reconcile Node 4 with node 3 to ensure that the replicas are synchronized.

Figure 42-5 Compare and Reconcile All Connected Replicas



42.1.4 Types of Replication Failover

This section describes the two types of replication failover types supported by Oracle Internet Directory.

There are two types of replication failover. They are:

- Stateless
- Time-based

Use stateless failover when you are unable to plan for the failover in advance. Stateless replication failover makes no assumptions about the state of the replicas. You can fail over to a new supplier at any time. Stateless failover requires more work after failover to synchronize the nodes.

Use time-based failover for planned failover. Time-based failover results in less work after failover. However, it requires some setup ahead of time to ensure that the following assumptions are true at the time of failover:

- The nodes are mostly synchronized
- The new supplier has preserved its change logs so that complete synchronization can be achieved quickly.

42.2 Performing a Stateless Replication Failover

This section explains how to perform a stateless replication failover.

This chapter consists of the following tasks:

- Stopping all Directory Replication Server on Related Nodes
- Breaking Old Replication Agreement and Setting up New Agreement
- Saving Last Change Number
- Comparing and Reconciling New Supplier and Consumer
- Updating Last Applied Change Number of New Agreement
- Cleaning Up Old Agreement on Old Supplier
- Starting All Directory Replication Server on Related Nodes

42.2.1 Stopping all Directory Replication Server on Related Nodes

Stop the Oracle directory replication servers on the new supplier, old supplier and consumer.

To stop all Directory replication server, execute the following command:

```
oidctl connect=connStr server=oidrepld instance=1 componentname=oidComponentName \
 flags="-h LdapHost -p LdapPort" stop
```

42.2.2 Breaking Old Replication Agreement and Setting up New Agreement

Break the old replication agreement between the old supplier and consumer and set up a new agreement between the new supplier and consumer.

Do this by using the Replication Environment Management Tool. Type:

```
remtool -pchgmaster [-v] [-bind consumer_host::port_number]
```

you are prompted for the *replication_dn_password*.

> **See Also:**
>
> The `remtool`command-line tool reference in *Reference for Oracle Identity Management*

42.2.3 Saving Last Change Number

Obtain the last change number from the new supplier.

For a one-way agreement, use the following command:

```
ldapsearch -h new_supplier_host -p port_number -b "" \
   -s base "objectclass=*" lastchangenumber
```

For a two-way agreement, use the following command:

```
ldapsearch -h consumer_host -p port_number -b "" \
   -s base "objectclass=*" lastchangenumber
```

Save this number!

42.2.4 Comparing and Reconciling New Supplier and Consumer

Use the Oracle Internet Directory Comparison and Reconciliation Tool to compare and reconcile the new supplier and consumer.

For a one-way agreement, type:

```
oidcmprec operation=reconcile \
   source=new_supplier_host:port \
   destination=consumer_host:port \
   base='""' scope=sub
```

For a two-way agreement, type:

```
oidcmprec operation=merge \
   source=new_supplier_host:port\
   destination=consumer_host:port/ \
   base='""' scope=sub
```

you are prompted for the source and destination replication dn passwords.

This example assumes that the entire directory is replicated and, therefore, that `base` is set to " ". If you are using partial replication, use the `base` and `dns2exclude` arguments to the `oidcmprec` tool to include the desired DIT.

> **See Also:**
>
> The `oidcmprec`command-line tool reference in *Reference for Oracle Identity Management*

42.2.5 Updating Last Applied Change Number of New Agreement

Modify the new agreement with the retrieved last applied number at the new supplier.

To do this:

1. Create an LDIF file with the last change number you retrieved in "Saving Last Change Number".

   For a one-way agreement, it should look similar to this:

   ```
   dn: agreement_dn
   changetype: modify
   replace: orclLastAppliedChangeNumber;apply$new_supplier_host$consumer_host
    orclLastAppliedChangeNumber;apply$new_supplier_host$consumer_host:
    last_change_number_retrieved.
   -
   replace: orclLastAppliedChangeNumber;transport$new_supplier_host$consumer_host
   orclLastAppliedChangeNumber;transport$new_supplier_host$consumer_host:
    last_change_number_retrieved_from_new_supplier
   ```

   For a two-way agreement, it should look similar to this:

   ```
   dn: agreement_dn
   changetype: modify
   replace: orclLastAppliedChangeNumber;apply$new_supplier_host$consumer_host
   orclLastAppliedChangeNumber;apply$new_supplier_host$consumer_host:
    last_change_number_retrieved_from_new_supplier
   -
   replace: orclLastAppliedChangeNumber;transport$new_supplier$consumer
   orclLastAppliedChangeNumber;transport$new_supplier$consumer:
    last_change_number_retrieved_from_new_supplier
   -
   replace: orclLastAppliedChangeNumber;apply$consumer_host$new_supplier_host
   orclLastAppliedChangeNumber;apply$consumer_host$new_supplier_host:
    last_change_number_retrieved_from_consumer
   -
   replace: orclLastAppliedChangeNumber;transport$consumer_host$new_supplier_host
   orclLastAppliedChangeNumber;transport$consumer_host$new_supplier_host:
    last_change_number_retrieved_from_consumer
   ```

2. Modify the agreement by using `ldapmodify`, as follows:

```
ldapmodify -D "cn=orcladmin" -q -h host_name -p port_number -f LDIF_file
```

42.2.6 Cleaning Up Old Agreement on Old Supplier

This section describes the procedure to clean up old agreement on old supplier by using the Replication Environment Management Tool.

If the old supplier was down when you performed Breaking Old Replication Agreement and Setting up New Agreement, the old agreement on the old supplier was not cleaned up. Clean it up now by using the Replication Environment Management Tool. Type:

```
remtool -pcleanup -agrmt [-v] [-bind consumer_host::port_number]
```

you are prompted for the *replication_dn_password*.

> **See Also:**
>
> The `remtool`command-line tool reference in *Reference for Oracle Identity Management*

42.2.7 Starting All Directory Replication Server on Related Nodes

This section describes the procedure to start the Oracle directory replication servers on the new supplier, the old supplier and the consumer.

To do this, type:

```
oidctl connect=connStr server=oidrepld instance=1 \
   name=instance_name componentname=component_name \
   flags="-h LdapHost -p LdapPort" start
```

> **See Also:**
>
> Understanding Process Control of Oracle Internet Directory Components

42.3 Performing a Time-Based Replication Failover

This section explains how to perform a time-based replication failover.

This section contains the following topics:

- Configuring Change Log Garbage Collection Object on New Supplier
- Saving Last Change Number from New Supplier
- Enabling Change Log Regeneration on New Supplier
- Waiting for the Desired Time Period to Elapse
- Stopping all Directory Replication Servers on Related Nodes
- Breaking Old Replication Agreement and Setting Up New Agreement
- Updating Last Applied Change Number of the New Agreement

- Cleaning Up Old Agreement on Old Supplier
- Starting All Directory Replication Servers on Related Nodes

42.3.1 Configuring Change Log Garbage Collection Object on New Supplier

Configure the changelog purging configuration entry on the new supplier so that it preserves change logs for the desired period.

For example, 24 hours, as follows:

1. Create an LDIF file similar to this:

```
dn: cn=changelog purgeconfig,cn=purgeconfig,cn=subconfigsubentry
changetype:modify
replace: orclpurgetargetage
orclpurgetargetage: 24
```

2. Apply the LDIF file by typing:

```
ldapmodify -D "cn=orcladmin" -q -p port -h host -f LDIF_file
```

> **See Also:**
>
> Managing Garbage Collection

42.3.2 Saving Last Change Number from New Supplier

Obtain the last change number from the new supplier.

Type the following command:

```
ldapsearch -h new_supplier_host -p port_number -D cn=orcladmin -q \
   -b "" -s base "objectclass=*" lastchangenumber
```

Save this number!

42.3.3 Enabling Change Log Regeneration on New Supplier

Enable change log regeneration at the new supplier, as follows:

1. Create an LDIF file like this:

```
dn:
changetype: modify
replace: orcldiprepository
orcldiprepository: TRUE
```

2. Apply the LDIF file by typing:

```
ldapmodify -D "cn=orcladmin" -q -h host_name -p port_number -f LDIF_file
```

42.3.4 Waiting for the Desired Time Period to Elapse

Wait for a period no greater than the value of `orclpurgetargetage` in the changelog purging configuration entry.

Wait for a period no greater than the value of `orclpurgetargetage` in the changelog purging configuration entry.

42.3.5 Stopping all Directory Replication Servers on Related Nodes

Stop the Oracle directory replication servers on the new supplier, old supplier and consumer.

Type the following command:

```
oidctl connect=connStr server=oidrepld instance=1 \
 componentname=oidComponentName \
 flags="-h LdapHost -p LdapPort" stop
```

> **See Also:**
>
> Understanding Process Control of Oracle Internet Directory Components

42.3.6 Breaking Old Replication Agreement and Setting Up New Agreement

Break the old replication agreement between the old supplier and the consumer, then set up a new agreement between the new supplier and the consumer.

Do this by using the Replication Environment Management Tool, as follows:

```
remtool -pchgmaster [-v] [-bind hostname:port_number]
```

you are prompted for the *replication_dn_password*.

> **See Also:**
>
> The `remtool`command-line tool reference in *Reference for Oracle Identity Management*

42.3.7 Updating Last Applied Change Number of the New Agreement

This section describes the procedure to Modify the new agreement at the new supplier.

Modify the new agreement at the new supplier so that its last applied change number has the value you retrieved in Saving Last Change Number from New Supplier, as follows:

1. Create an LDIF file with the retrieved last applied change number, similar to this:

   ```
   dn: agreement_dn
   changetype: modify
   replace: orclLastAppliedChangeNumber
   orclLastAppliedChangeNumber: last_change_number_retrieved
   ```

2. Apply the LDIF file to the agreement by using `ldapmodify`:

   ```
   ldapmodify -D "cn=orcladmin" -q -h host_name -p port_number -f LDIF_file
   ```

42.3.8 Cleaning Up Old Agreement on Old Supplier

You can clean up old agreement on old supplier node by using the Replication Environment Management Tool.

If the old supplier was down when you performed "Breaking Old Replication Agreement and Setting Up New Agreement", the old agreement on the old supplier was not cleaned up. Clean it up now by using the Replication Environment Management Tool. Type:

```
remtool -pcleanup -agrmt [-v] [-bind hostname:port_number]
```

you are prompted for the *replication_dn_password*.

> **See Also:**
>
> The `remtool` command-line tool reference in *Reference for Oracle Identity Management*

42.3.9 Starting All Directory Replication Servers on Related Nodes

This section describes the procedure to start the Oracle directory replication servers on the new supplier, the old supplier and the consumer.

Type the following command:

```
oidctl connect=connStr server=oidrepld instance=1  \
 componentname=oidComponentName \
 flags="-h LdapHost -p LdapPort" start
```

> **See Also:**
>
> - Understanding Process Control of Oracle Internet Directory Components
> - The `oidctl`command-line tool reference in *Reference for Oracle Identity Management*

43
Managing Replication Configuration Attributes

The following topics describe the configuration attributes that control the Oracle Internet Directory replication server and how to manage these attributes using LDAP command-line utilities:

- Understanding Replication Configuration Attributes
- Managing Replication Configuration Attributes Using the Command Line

See Managing and Monitoring Replication for specific replication management tasks.

43.1 Understanding Replication Configuration Attributes

The following topics provide a contextual description of the configuration attributes that reside in specific containers in the DIT:

- Replication Configuration Container
- Understanding Replica Subentry
- Understanding Replication Agreement Entry
- Replication Naming Context Container Entry
- Understanding Replication Naming Context Object Entry
- Understanding Replication Configuration Set
- Examples of Replication Configuration Objects in a Directory

43.1.1 Replication Configuration Container

All replication information for a node resides in the container `cn=replication configuration` located at the root DSE. This entry resides on each node in a DRG.

The following is a sample replication configuration container entry:

```
dn: cn=replication configuration
orclaci: access to entry by * (browse)
orclaci: access to attr=(*) by * (search,read)
orclnormdn: cn=replication configuration
cn: replication configuration
description: Replication agreement Container object
objectclass: top
objectclass: orclcontainerOC
```

43.1.2 Understanding Replica Subentry

The following topics provide a contextual description of replica subentry and its attributes:

- About Replica Subentry
- Replica Subentry Attributes
- Example of Replica Subentry

43.1.2.1 About Replica Subentry

The Replica subentry has the DN

`orclreplicaid=Replica_ID,cn=replication configuration`

This subentry is created at installation under the replication configuration container. It contains attributes that identify and define the characteristics of the node it represents.

43.1.2.2 Replica Subentry Attributes

Table 43-1 describes the attributes of the replica subentry. LDAP indicates that you can manage this attribute by using LDAP tools.

Table 43-1 Attributes of the Replica Subentry

Attribute	Description	Update Mechanism	Default	Possible Values
`orclreplicaid`	Unique identifier for directory database. Initialized at installation. Matches orclreplicaid at the root DSE.	Read-only	*hostname_ORACLESID*	Integer
`orclreplicauri`	Address used to open a connection to this replica.	LDAP		Valid ldapURI format
`orclreplicasecondaryuri`	Addresses used if `orclReplicaURI` cannot be used.	LDAP		Valid ldapURI format
`orclreplicatype`	Defines the type of replica such as read-only or read/write.	LDAP	0 (Read/Write)	0: Read/Write 1: Read-Only 2: Pilot
`orclpilotmode`	Defines whether replica is in pilot (testing) mode.	`remtool -pilotreplica`	0	0: False 1: True
`orclreplicastate`	Defines state of the replica.	LDAP		You can set 0, 1, 2, 6, or 8. Server sets other values. See Table A-2.
`pilotstarttime`	Time when replica entered pilot (testing) mode.	Read-only		Time

> **Note:**
>
> On Windows systems, ensure that the replication server is not running before you enable bootstrapping by changing the value of `orclReplicaState` to `0`.

43.1.2.3 Example of Replica Subentry

In Figure 43-3 , a replica subentry is represented by `orclReplicaID=UID_of_node_D,cn=replication configuration`.

The following is a sample replica subentry:

```
dn: orclreplicaid=myhost1_repl1,cn=replication configuration
objectclass: top
objectclass: orclreplicasubentry
orclreplicaid: myhost1_repl1
orclreplicauri: ldap://myhost1:3060/
orclreplicasecondaryuri: ldap://myhost1.mycompany.com:3060/
orclreplicastate: 1
```

> **See Also:**
>
> Oracle Directory Replication Schema Elements in *Reference for Oracle Identity Management* for descriptions of the attributes of the replica subentry.

43.1.3 Understanding Replication Agreement Entry

The following topics provide a conceptual description of replication agreement entry and its attributes and also describe the different LDAP replication agreements:

- About Replication Agreement Entry
- Replication Agreement Entry Attributes
- About LDAP-Based Replication Agreements
- Example of Two-Way LDAP-Based Replication Agreements

43.1.3.1 About Replication Agreement Entry

The DN of the Replication Agreement Entry is:

`orclagreementid=`*`Agreement_ID`*`,orclreplicaid=`*`Replica_ID`*`,cn=replication configuration`

This entry contains attributes that define the replication agreement between the two or more nodes and is associated with the `orclReplAgreementEntry` objectclass. LDAP-based replication agreement for LDAP nodes reside under the supplier's replica subentry. For example, in Figure 43-3, an LDAP-based replication agreement entry is represented by `orclagreementID=000003, orclReplicaID=UID_of_node_D,cn=replication configuration`.

43.1.3.2 Replication Agreement Entry Attributes

Table 43-2 shows the attributes in the replication agreement. LDAP indicates that you can manage this attribute by using LDAP tools.

Table 43-2 Attributes of the Replication Agreement Entry

Attribute	Description	Update Mechanism	Default	Possible Values
`orclagreementid`	Name of replication agreement entry.	Read-only		
`orclreplicadn`	LDAP-based replication only. DN of the replica to identify a consumer in the replication agreement.	Read-only		DN

Table 43-2 (Cont.) Attributes of the Replication Agreement Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclentryexclusionfilter	LDAP-based one-way replication only. LDAP filter string that specifies entries that should not be replicated. See Configuring Replication Filtering Using the orclEntryExclusionFilter Attribute.	LDAP		LDAP filter string enclosed in parentheses. Supported maximum length is 1500 characters. For example: (sn=smith)
orclreplicationprotocol	Replication protocol for change propagation to replica. Values:	Read-only		ODS_LDAP_1.0: LDAP-based
orclupdateschedule	Update interval for new changes and those being retried.	LDAP	60 (seconds)	Greater than or = 0
orclhiqschedule	Interval at which the directory replication server repeats the change application process.	LDAP	600 (seconds)	Greater than or = 60 (seconds)
orclldapconnkeepalive	Whether the connections from replication server to the directory server are kept active or established every time the changelog processing is done.	LDAP	1	0: false 1: true
orcllastappliedchangenumber	Last change number transported or applied at the consumer replica. For LDAP-based agreements, this attribute contains subtypes. The format is: orcllastappliedchangenumber; *status_type*$*supplier_replicaID* $*consumer_replicaID*: Number where *status_type* is: transport or apply, *supplier_replicaID* and *consumer_replicaID* indicate the direction of LDAP data flow, and Number is the last applied change number. It indicates that changelogs from *supplier_replicaID* to *consumer_replicaID* with change number less than Number have been transported/applied at node *consumer_replicaID*.	Read-only		
orclexcludednamingcontexts	Subtrees to be excluded from replication.	Read-only		

Table 43-2 (Cont.) Attributes of the Replication Agreement Entry

Attribute	Description	Update Mechanism	Default	Possible Values
orclreplicationid	Unique identifier of a one-way, two-way, or peer-to-peer replication group	Read-only		
orclagreementtype	Replication agreement type.	Read-only		0: one-way/read-only fan-out replication agreement 1: two-way/ updatable fan-out replication agreement 2: LDAP-based multimaster replication agreement

43.1.3.3 About LDAP-Based Replication Agreements

For LDAP-based replication, there are separate replication agreements for each supplier-consumer relationship. For one-way replication, there is a single, one-way replication agreement.

The entry for an LDAP-based replication agreement resides immediately below the replica subentry of the node that serves as the supplier. Thus, the DN of the replication agreement as found on a supplier node is:

```
orclagreementID=unique_identifier_of_the_replication_agreement,
orclReplicaID=unique_identifier_of_supplier_node, cn=replication configuration
```

Similarly, the DN of the replication agreement as found on a consumer node is:

```
orclagreementID=unique_identifier_of_the_replication_agreeement,
orclReplicaID=unique_identifier_of_supplier_node, cn=replication configuration
```

In a fan-out replication agreement, you can tell which node the agreement entry is associated with by looking at its parent. For example, look at the following replication agreement entry.

```
orclagreementID=000002,orclReplicaID=node_A,cn=replication configuration
```

In this example, you can determine that the replication agreement represented by `orclagreementID=000002` is associated with node A. This is because the parent of `orclagreementID=000002` is `orclReplicaID=node_A`.

Note:

- The container entry `cn=replication configuration` is replicated on all nodes, but may not be identical on all nodes.
- The `orclreplicadn` attribute of an LDAP-based replication agreement specifies the associated consumer node.
- The `agreementtype` indicates the replication agreement type. See Table 43-2 for values of `orclagreementtype`.

43.1.3.4 Example of Two-Way LDAP-Based Replication Agreements

For two-way replication, there can be either a single, two-way replication agreement or two one-way agreements for each supplier-consumer relationship. The following is a sample two-way replication agreement entry:

```
dn: orclagreementid=000002, orclreplicaid=stadd58_repl, cn=replication configuration
orclagreementid: 000002
orclreplicationprotocol: ODS_LDAP_1.0
orclreplicadn: orclreplicaid=stadd57_repl,cn=replication configuration
orclldapconnkeepalive: 1
orclagreementtype: 1
orclreplicationid: 000002
orcllastappliedchangenumber;transport$stadd57$stadd58: 106
orcllastappliedchangenumber;transport$stadd58$stadd57: 2421
orcllastappliedchangenumber;apply$stadd57$stadd58: 106
orcllastappliedchangenumber;apply$stadd58$stadd57: 2421
orclupdateschedule: 0
orclhiqschedule: 60
objectclass: orclReplAgreementEntry
objectclass: top
```

Note:

The value of `orclagreementtype` is 1 because this is a two-way replication agreement. See Table 43-2 for values of `orclagreementtype` for other replication agreement types.

See Also:

Oracle Directory Replication Schema Elementsin *Reference for Oracle Identity Management* for descriptions of the attributes of the replication agreement entry

43.1.4 Replication Naming Context Container Entry

This entry contains all the LDAP naming context objects. This entry has the RDN `cn=replication namecontext`, and it is created below the `orclagreementID` entry during replication configuration.

The following is a sample replication naming context container entry:

```
dn: cn=replication namecontext,orclagreementid=000002,
 orclreplicaid=myhost1_repl1,cn=replication configuration
objectclass: top
objectclass: orclcontainerOC
cn: replication namecontext
```

43.1.5 Understanding Replication Naming Context Object Entry

The following topics provide a contextual description of replication naming context object entry and its attributes:

- About Replication Naming Context Object Entry
- Replication Naming Context Entry Attributes
- Example of Replication Naming Context Entry

43.1.5.1 About Replication Naming Context Object Entry

This entry contains all the LDAP naming context objects. These objects specify the replication filtering policy, that is, what to include in or exclude from replication to an LDAP-based partial replica.

This entry is created below the naming context container entry during replication configuration. It is configurable. For example, in Figure 43-3, the replication naming context object is: `cn=includednamingcontext000001,cn=replication namecontext,orclagreementID=000003,orclReplicaID=`*`UID_of_node_D`*`,cn=replication configuration`.

43.1.5.2 Replication Naming Context Entry Attributes

Table 43-3 describes the attributes of the replication naming context entry.

Table 43-3 Attributes of the Replication Naming Context Entry

Attribute	Description
orclincludednamingcontexts	The root of the naming context to be replicated. If orclincludednamingcontexts is set to "*", all the naming contexts are replicated. This attribute has subtypes that specify the replication direction in which the naming context should be included. The format is: orclincludednamingcontexts ; *supplier_replicaID*$consumer_replicaiD: DN This is a single valued attribute. For each naming context object, you can specify only one unique subtree in each direction. In partial replication, except for subtrees listed in the orclexcluednamingcontexts attribute, all subtrees in the specified included naming context are replicated. You can modify this attribute.
orclexcludednamingcontexts	The root of a subtree, located within the included naming context, to be excluded from replication.This attribute has subtypes that specify the replication direction in which the naming context should be excluded. The format is: orclexcludednamingcontexts; *supplier_replicaID$consumer_replicaiD* : DN This is a multivalued attribute. From within the naming context specified in the orclincludednamingcontexts attribute, you can specify one or more subtrees to be excluded from the partial replication in each direction. You can modify this attribute.
orclexcludedattributes	An attribute, located within the included naming context, to be excluded from replication. Orclexcludedattributes has subtypes that specify the replication direction in which the specified attribute should be excluded. The format is: orclexcludedattributes; *supplier_replicaID$consumer_replicaiD*: attribute_name This is a multivalued attribute. You can modify this attribute

43.1.5.3 Example of Replication Naming Context Entry

The following is a sample replication naming context object entry:

```
dn:cn=namectx001,
cn=replication namecontext,
orclagreementid=unique_identifier_of_the_replication_agreement,
orclreplicaid=replica_id_of_node_A,
cn=replication configuration
orclincludednamingcontexts: cn=mycompany
orclexcludednamingcontexts; replica_id_of_node_A$ replica_id_of_node_B :
c=us,cn=mycompany
orclexcludedattributes; replica_id_of_node_B$ replica_id_of_node_A : userPassword
```

The example specifies the following replication filtering:

- The naming context cn=mycompany is included for replication in both directions for node A and node B.

- The naming context `c=us,cn=mycompany` is excluded for replication from node A to node B only.
- The `userPassword` attribute is excluded for replication from node B to node A

> **See Also:**
>
> - Scenarios of Filtering Naming Context in LDAP Replication
> - Oracle Directory Replication Schema Elementsin *Reference for Oracle Identity Management*

43.1.6 Understanding Replication Configuration Set

The following topics provide a a contextual description of replication configuration set and its attributes:

- About Replication Configuration Set
- Replication Configuration Set Attributes

43.1.6.1 About Replication Configuration Set

The replication configuration set has the DN:

```
cn=configset0,cn=osdrepld,cn=subconfigsubentry
```

Table 43-4 lists and describes the attributes of the replication configuration set, which has the following DN:

```
cn=configset0,cn=osdrepld,cn=subconfigsubentry
```

You must restart the replication server in order for any attribute changes under this DN to take effect, except for the attribute `orcldebuglevel`. LDAP indicates that you can manage this attribute by using LDAP tools.

43.1.6.2 Replication Configuration Set Attributes

The following table lists and describes the attributes of the replication configuration set:

Table 43-4 Replication Configuration Set Attributes

Attribute	Description	Update Mechanism	Default	Possible Values
`modifyTimestamp`	Time of entry creation or modification.	Read-only		
`modifiersName`	Name of person creating or modifying the entry	Read-only		
`orclChangeRetryCount`	Number of processing retry attempts for a change-entry before being moved to the human intervention queue.	LDAP	10	Greater than or equal to 1.

Table 43-4 (Cont.) Replication Configuration Set Attributes

Attribute	Description	Update Mechanism	Default	Possible Values
orclThreadsPerSupplier;transport	Number of worker threads spawned at each supplier for transporting change logs.	LDAP	1	1-100
orclThreadsPerSupplier;apply	Number of worker threads spawned at each supplier for applying change logs.	LDAP	5	1-100
orclreplautotune	Dynamically vary the number of threads assigned to transport and apply tasks based on load. If you set the server to auto tune, you must specify the number of maximum number of threads to be shared between these tasks. Restart server after changing.	LDAP	1	0: Off 1: On
orclreplmaxworkers	Maximum number of worker threads. Required if orclreplautotune is set.	LDAP	20	1-100
orclsdumpflag	Generate stack dump. (Restart after changing.)	LDAP	0	0: False 1: True
orclmaxlogfilesize	Maximum log file size (MB)	LDAP	1 MB	> or = 1
orclmaxlogfiles	Maximum number of log files to keep in rotation	LDAP	100	> or = 1
orclsizelimit	Maximum number of entries to process per replication cycle	LDAP	1000	1-10000
orclconflresolution	Automatically resolve replication conflicts	LDAP	1	0: False 1: True
orclreplusesasl;digest-md5	Use SASL for replication binds.	LDAP	Attribute does not exist by default.	auth, auth-int, auth-conf
orclActivateReplication	Specifies that replication be activated on the replication server designated by orclOidInstanceName and orclOidComponentName.	LDAP	0	0: False 1: True
orclReplicationState	Activation state of the replication server	Read-only, LDAP	0	0 or nonexistant: Not running False 1: Running

Table 43-4 (Cont.) Replication Configuration Set Attributes

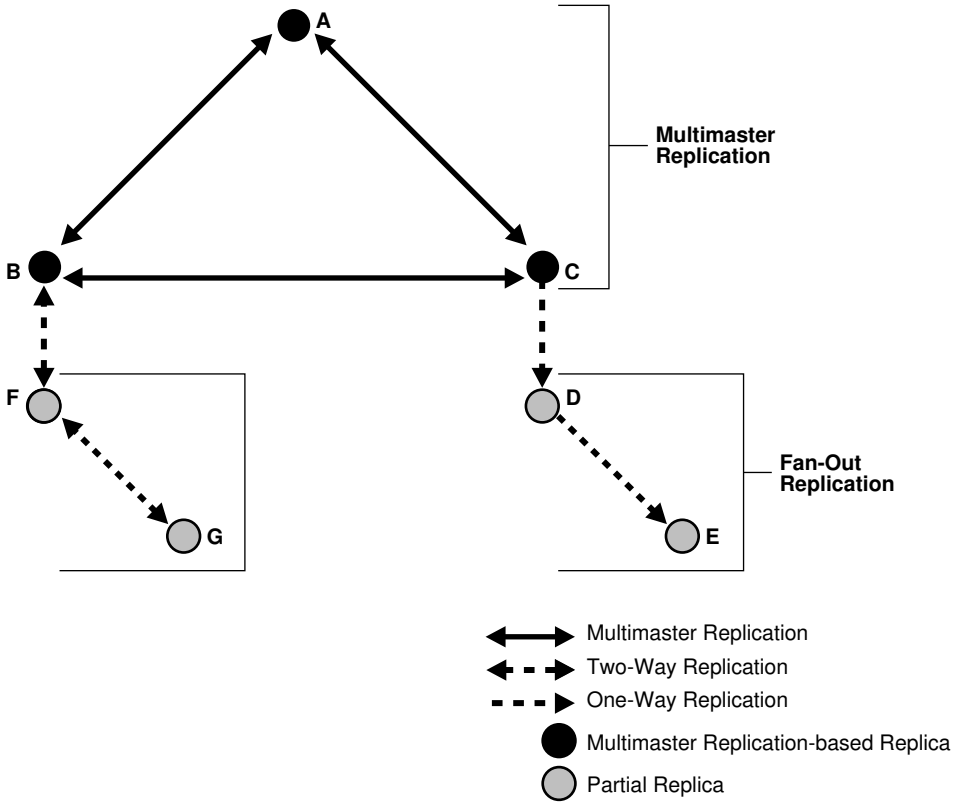
Attribute	Description	Update Mechanism	Default	Possible Values
orcldebuglevel	Debug level of replication server	LDAP	0	Values are additive: 0: No Debug Log 2097152: Replication Performance Log 4194304: Replication Debug Log 8388608: Function Call Trace 16777216: Heavy Trace Log
orclOidComponentName	Name of OID component where replication is or will be activated	Read-only	Set during replication setup	String
orclOidInstanceName	Instance number of instance where replication is or will be activated	Read-only	Set during replication setup	Integer
orclReplAttrConflict	Specifies whether timestamp or attribute version should be honored first during attribute level conflict resolution.	LDAP	0	0: Timestamp first. 1: Version number first

43.1.7 Examples of Replication Configuration Objects in a Directory

This section explains the replication configuration objects in a directory.

The examples of replication objects in this section rely on the replication environment shown in Figure 43-1.

Figure 43-1 Example: Multimaster Replication and Fan-Out Replication



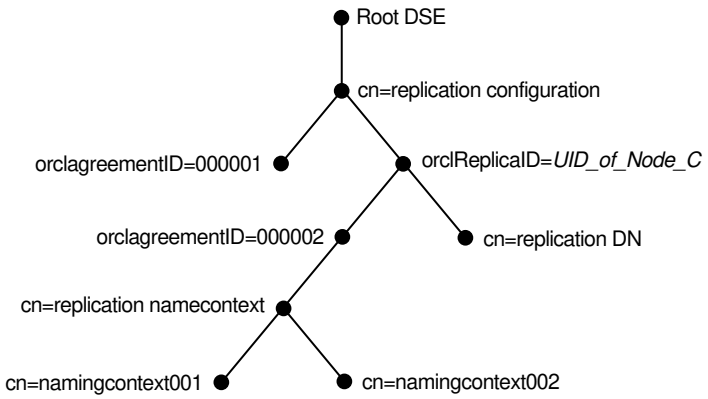
In Figure 43-1, nodes A, B, and C form a multimaster replication group. Node C replicates to a fourth node, D, which, in turn, fans out to Node E.

The replication agreements in this environment are as follows:

- Node A has one replication agreement representing its multimaster relationship with nodes B and C.
- Node B has two replication agreements, the first representing its multimaster relationship with nodes A and C, the second representing its relationship to node F. The replication agreement between B and F is two-way.
- Node C has two replication agreements, the first representing its multimaster relationship with nodes A and B, the second representing its relationship to node D. This is a one-way replication agreement in which C serves as the supplier and node D is the consumer.
- Node D has two replication agreements. Both of its replication agreements are one-way. One represents its relationship to the supplier node C, from which it consumes changes, the other represents its relationship to consumer node E for which it is the supplier.
- Node E has a one-way replication agreement with Node D. Node E is the consumer.
- Node F has two replication agreements, one representing its relationship to the node B, the other representing its relationship to node G.Both are two-way replication agreements.
- Node G has a one-way replication agreement with Node F. Node G is the consumer.

Figure 43-2 shows the replication objects in the DIT that pertain to node C in Figure 43-1 .

Figure 43-2 Example: Replication Configuration Entries for Node C



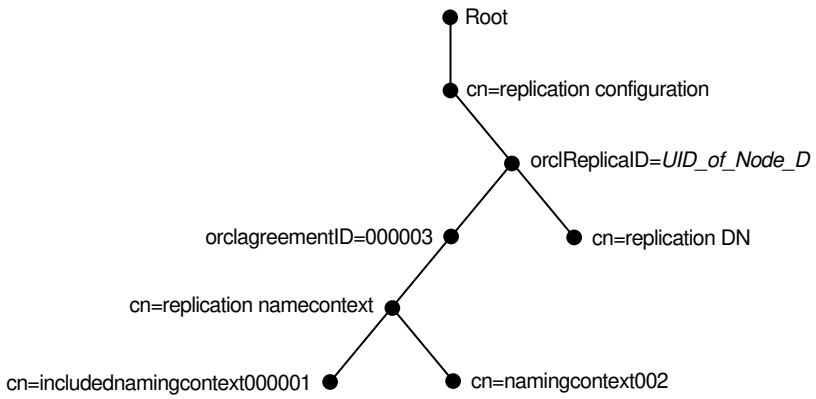
For node C, the entry `cn=replication configuration` at the root DSE contains these RDNs:

- `orclagreementID=000001`: The multimaster replication agreement in which node C participates with nodes A and B.
- `orclReplicaID=`*`UID_of_node_C`*: Unique identifier of node C that contains information about it.
- `orclagreementID=000002`: Unique identifier of the relationship between supplier node C and consumer node D. You know that, in this case, `orclagreementID=000002` is the replication agreement of the supplier node C because node C is its parent.

  This entry contains the attribute `orclreplicaDN`. Its value is the replica entry DN of consumer node D, with which node C has the replication agreement.
- `cn=replication DN`: The bind DN that the directory replication server on node C uses to bind to the directory server.
- `cn=replication namecontext`: Container of information about naming contexts that are included in replication.
- `cn=includednamingcontext000001` and `cn=namingcontext002`: The actual objects that are included in or excluded from replication. In the naming context included for replication, you can specify one or more subtrees to be excluded from replication. In that same included naming context, you can specify particular attributes to be excluded from replication.

Figure 43-3 shows the replication agreement entry in the DIT that pertains to node D in Figure 43-1.

Figure 43-3 Example: Replication Configuration Entries for Node D



For node D, the entry `cn=replication configuration` at the root DSE contains these RDNs:

- `orclReplicaID=`*`UID_of_node_D`*: Unique identifier of node D and contains information about it.
- `orclagreementID=000003`: Unique identifier of the relationship between supplier node D and consumer node E. You know that, in this case, `orclagreementID=000003` is the replication agreement of the supplier node D because node D is its parent.

  This entry contains the attribute `orclreplicaDN`, the value of which is the DN of consumer node E with which node D has the replication agreement.
- `cn=replication DN`: Bind DN that the directory replication server on node D uses to bind to the directory server.
- `cn=replication namecontext`: Container of information about naming contexts that are included in replication.
- cn=`namingcontext001` and `cn=namingcontext002`: Objects specifying naming contexts to be included in replication. In the naming context included in replication, you can specify one or more subtrees or particular attributes to be excluded from replication.

43.2 Managing Replication Configuration Attributes Using the Command Line

You can modify most attributes from the command line by using `ldapmodify`.

The command line syntax is:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

The contents of the LDIF file depends on the DN and the operation being performed.

For examples of LDIF files for changing replication configuration attributes, see Overview of Managing and Monitoring Replication Using the Command Line.

44

Managing and Monitoring Replication

This chapter describes how to monitor and manage replication in Oracle Internet Directory using Oracle Directory Services Manager (ODSM) and LDAP command-line utilities. It also describes how to compare and reconcile inconsistent data using the Oracle Internet Directory Comparison and Reconciliation Tool (`oidcmpre`).
For more information about replication configuration attributes, see Managing Replication Configuration Attributes.

This chapter includes the following sections:

- Introduction to Managing and Monitoring Replication
- Managing and Monitoring Replication by Using ODSM
- Overview of Managing and Monitoring Replication Using the Command Line
- Overview of Comparing and Reconciling Inconsistent Data Using oidcmprec

> **Note:**
>
> Some changes do not take effect until the replication server is restarted.

44.1 Introduction to Managing and Monitoring Replication

After you have installed and configured replication, you can view or modify the default values for replication-related attributes.

The attributes and their containers are described in Managing Replication Configuration Attributes.

> **See Also:**
>
> - Understanding Oracle Internet Directory Replication
> - Setting Up Replication
> - How Replication Works

This introduction includes the following topics:

- Implications of LDAP-Based Partial Replication
- About Managing Worker Threads
- Change Logs in Directory Replication
- Overview of Change Log Partitioning in Directory Replication
- The Human Intervention Queue

- About Pilot Mode
- Overview of Conflict Resolution in Oracle Replication

44.1.1 Implications of LDAP-Based Partial Replication

In LDAP-based partial replication, you can change what is or is not replicated by modifying replica naming context objects. The parameters for these objects are stored in entries that have this DN:

For example:

```
cn=namingcontext_ID,cn=replication namecontext,
 orclAgreementID=numeric_identifier_of_replication_agreement,
 orclReplicaId=unique_identifier_of_replica, cn=replication configuration
```

> **Note:**
>
> Because the directory replication server reads replica naming context objects from the agreement located at the supplier, you must apply all modifications against naming context objects at the supplier and, optionally, at the consumer.

See Overview of Modifying Replica Naming Context Object Parameters Using ldapmodify.

44.1.2 About Managing Worker Threads

The replication server is a multithreaded process.

You can control the number of worker threads per supplier that are dedicated to:

- Transporting the changelogs from supplier to consumer
- Applying the transported changelogs at consumer

You can set the number of transport threads and the number of apply threads per supplier. Alternatively, you can configure the replication server to auto tune, that is, to dynamically vary the number of threads assigned to these two tasks based on load. If you set the server to auto tune, you must specify the number of maximum number of threads to be shared between these tasks.

You must tune these numbers based on load. From the command line, you use `ldapmodify` to configure threads.

> **See Also:**
>
> - Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify
> - Table 43-4

44.1.3 Change Logs in Directory Replication

Oracle Internet Directory records each change as an entry in the change log store. Each entry has a unique change number.

The consumer keeps track of the change number of the last change it applied, and it retrieves from the supplier only those changes with numbers greater than that of the last change it applied.

- In an LDAP-based replication agreement, change log processing consists of two phases, transporting the change log and applying the change log. For each LDAP-based agreement, there are two change log processing statuses, one for the each phase. The directory replication server stores the last change number it transported in the `transport` subtype of the `orlcllastappliedchangenumber` attribute of the replication agreement entry. The directory replication server stores the last change number it applied in the `apply` subtype of the `orllclastappliedchangenumber` attribute of the replication agreement entry. The format of the `orlcllastappliedchangenumber` attribute is shown in Table 43-2.

  > **Note:**
  >
  > Beginning with Oracle Internet Directory 11g Release 1 (11.1.1.7.0), the data flow in LDAP replication consists of the apply phase with the apply queue. The transport phase with the transport queue is no longer used. See Architecture of LDAP-Based Replication.

- When a multivalued attribute is modified, Oracle Internet Directory normally generates a change log containing all the values of that attribute, even if the modification added only one new value. However, for `member` and `uniquemember`, which might contain millions of values, this behavior would cause performance issues. Instead, for `member` and `uniquemember`, Oracle Internet Directory generates a change log containing only the values added or deleted. This behavior is controlled by the DSE attribute `orclsimplemodchglogattributes`.

Change logs are purged by the garbage collector after they have been consumed by the replication server.

44.1.4 Overview of Change Log Partitioning in Directory Replication

Oracle Internet Directory records each change as an entry in the change log table.

This section includes the following topics:

- About Change Log Partitioning in Directory Replication
- Change Log Partitioning Strategy
- Configuring Change Log Partitioning

44.1.4.1 About Change Log Partitioning in Directory Replication

The change log tables routinely contain few thousands to millions of records. However, such high volume tables have key considerations, for instance:

- Ongoing updates resulting in addition of data into these tables.

- Ongoing replication search for new change records.
- Periodic purge to remove obsolete data.

To address these needs, Oracle Internet Directory introduces the concept of partitioning of change log tables. Partitioning allows you to break large tables into smaller and more manageable pieces called partitions. Each partition has its own name, computing resources, and storage characteristics. When a query is executed, the request is forwarded to the partition that holds the row that needs to be processed. This improves the performance of query execution, enhances the efficiency, and simplifies the day-to-day maintenance complexities.

> **Note:**
>
> Change log partitioning is supported on Oracle Database 11*g* Enterprise Edition or a higher version.

44.1.4.2 Change Log Partitioning Strategy

Oracle Internet Directory allows you to partition the change log table, `ODS_CHG_LOG`, by defining partition range of `100`K. This implies that each partition will have 100K records, because the database creates a new partition for every 100K records.

When a purge is performed, it drops the partition instead of individual records. This in turn implies that not all changes are purged. Purge takes place only when maximum numbers for change logs reach certain interval value.

44.1.4.3 Configuring Change Log Partitioning

To configure partitioning for change log tables, you need to explicitly run the *oidchlogs.sql* script after upgrading or installing the product. This SQL script is shipped with the installation package.

> **Note:**
>
> - Default installation of Oracle Internet Directory does not enable change log partitioning.
> - You must stop all Oracle Internet Directory instances before running the *oidchlogs.sql* script for change log partitioning.

Navigate to the following directory to run the SQL script as Oracle Internet Directory database user while the service is down:

`$ORACLE_HOME/ldap/admin/oidchlogs.sql`

On successful partitioning, a table by the name `chglog_[timestamp]` is created as a backup partitioned `ods_chg_log` table.

44.1.5 The Human Intervention Queue

This section describes the roles of the human intervention queue, the purge queue, and the retry queue in replication.

Managing an Entry Using Multimaster Replication Process describes the roles of the human intervention queue, the purge queue, and the retry queue in replication. Overview of Conflict Resolution in Oracle Replication provides information about the role of these queues in conflict resolution.

44.1.5.1 About Managing the Queues

The human intervention queue tools, `ManageHiq.retry` and `ManageHiq.purge`, enable you to move changes from the human intervention queue to the retry queue or the purge queue, respectively. See Managing the Human Intervention Queue .

44.1.5.2 About Queue Statistics

You can view queue statistics by using the command line. See Overview of Viewing Change Logs Using ldapsearch.

44.1.5.3 The Number of Entries the Human Intervention Queue Tools Can Process

If the number of entries in the Human Intervention Queue is greater than the maximum number of changelogs the replication server can process at a time, some entries are never processed.

The maximum number of changelogs the replication server can process at a time is the minimum of two configuration attributes:

- `orclsizelimit` in the replication configuration set
- `orclsizelimit` in the instance-specific configuration entry of the OID component where replication is active

The default value of `orclsizelimit` in the replication configuration set is 1000. If you set it to 0, it takes the default value of 1000.

The `orclsizelimit` attribute in the Oracle Internet Directory instance-specific entry specifies the maximum number of entries to be returned by a search. (Its default value is 10000.

To increase the number of changelogs processed at a time, you must set both attributes to the same value, a value greater than 1000.

> **See Also:**
>
> - Managing the Number of Entries the Human Intervention Queue Tools Can Process
> - Table 43-4

Setting the Oracle Internet Directory server instance attribute `orclsizelimit` very high impacts server performance, because `orclsizelimit` in the Oracle Internet Directory server instance also controls the maximum number of entries to be returned by a search.

44.1.6 About Pilot Mode

Pilot mode is used to test an application before deploying it in production.

Typically, you set pilot mode on the local node, with one-way replication from the production node. While the replica is in pilot mode, all the LDAP changes occurring at the local node are tracked. When you end pilot mode, those changes are written to an LDIF file. When the application is deployed in production, all the entries added or modified in the pilot replica can be added to the production node using the ldif file.

> **See Also:**
>
> Specifying Pilot Mode for a Replica by Using remtool

44.1.7 Overview of Conflict Resolution in Oracle Replication

Conflicts occur whenever the directory replication server attempts to apply remote changes from a supplier to a consumer and, for some reason, fails.

This section explains the various types of replication conflicts, about the automated conflict resolution, and how it works.

This section includes the following topics:

- About Conflict Resolution in Oracle Replication
- Levels at Which Replication Conflicts Occur
- Resolving Replication Conflicts Automatically
- How Automated Conflict Resolution Works

44.1.7.1 About Conflict Resolution in Oracle Replication

Two-way and multimaster LDAP-based replication enable updates to multiple directory servers. Conflicts occur whenever the directory replication server attempts to apply remote changes from a supplier to a consumer and, for some reason, fails.

Conflicts usually stem from differences in the timing of changes arising from the occasional slowness or transmission failure over wide area networks. Also, an earlier inconsistency might continue to cause conflicts if it is not resolved in a timely manner.

In partial replication, when a naming context is changed from included to excluded, the replication server deletes the naming context at the consumer. Similarly, if the naming context is changed from excluded to included, the replication server synchronizes the entire naming context from supplier to consumer.

LDAP operations that can lead to conflicts include:

- Addition
- Deletion
- Modification
- Modification of either an RDN or a DN

44.1.7.2 Levels at Which Replication Conflicts Occur

There are two types of conflicts:

- Entry-level conflicts

- Attribute-level conflicts

Table 44-1 Types of Replication Conflict

Level of Replication Conflict	Description
Entry-level conflicts	Caused when the directory replication server attempts to apply a change to the consumer. One of the following types of changes to the consumer could occur: • Adding an entry that already exists • Deleting an entry that does not exist • Modifying an entry that does not exist • Applying a modifyrdn operation when the DN does not exist These conflicts can be difficult to resolve. For instance, it may be impossible to resolve a conflict because: • The entry has been moved to a different location • The entry has not yet arrived from a supplier • The entry has been deleted • The entry never existed on the consumer If an entry exists and it should not, then it may be because it was added earlier, or that it recently underwent a modifydn operation.
Attribute-level conflicts	Caused when two directories are updating the same attribute with different values at different times. If the attribute is single-valued, then the replication process resolves the conflict by examining the timestamps of the changes involved in the conflict and the attribute version number. The attribute `orclReplAttrConfl` in the replication configuration set entry determines which is honored first. If `orclReplAttrConfl` is `0` (the default) timestamp is honored first. If it is `1`, attribute version is honored first.

44.1.7.3 Resolving Replication Conflicts Automatically

You can enable automatic conflict resolution. When this feature is enabled, conflicts in the Human Intervention Queue are automatically moved to the purge queue if the supplier's schema and consumer's schema match.

You can use the `ldapmodify` command to enable or disable automatic conflict resolution.

To use the command line, change the value of `orclconflresolution` in the replication configuration set.

> **See Also:**
>
> - Table 43-4
> - Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify

44.1.7.4 How Automated Conflict Resolution Works

The directory replication server attempts to resolve all conflicts that it encounters by following this process:

1. The conflict is detected when a change is applied.

2. The replication process attempts to reapply the change a specific number of times or repetitively for a specific amount of time after a specific waiting period.
3. If the replication process reaches the retry limit without successfully applying the change, it flags the change as a conflict, which it then tries to resolve. If the conflict cannot be resolved according to the resolution rules (described in the next section), the change is moved to a low-priority, human intervention queue. Changes are then applied according to the time unit specified in the `orclHIQSchedule` parameter in the replication agreement. Before it moves the change, the directory replication server writes the conflict into a log file for the system administrator.

> **Note:**
>
> There is no conflict resolution of schema, catalog, and group entries during replication. This is because attempting resolution of such large multivalued attributes would have a significant negative impact on performance. Be careful to avoid updating such entries from more than one master at a time.

> **See Also:**
>
> - How Replication Works for descriptions of how the multimaster replication process adds, deletes, and modifies entries, and how it modifies DNs and RDNs.
> - LDAP Schema Overview in *Reference for Oracle Identity Management*for schema questions
> - `catalog` command-line tool reference in *Reference for Oracle Identity Management* for catalog questions
> - Section on managing group entries in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager* for group entry questions.

44.2 Managing and Monitoring Replication by Using ODSM

You can manage and monitor replication by using ODSM.

See Managing Local Change Logs by Using Oracle Directory Services Manager.

44.2.1 Managing Local Change Logs by Using Oracle Directory Services Manager

Oracle Directory Services Manager enables you to view the last 500 changes you performed, listing them by change log number, the type of operation—namely, add, modify, or delete—in which each occurred, and the entry on which each was made. It allows you select a particular change to see more specific details about it.

This section contains the following topics:

- Viewing Local Change Logs Using ODSM
- Properties of the Change Log Entry

44.2.1.1 Viewing Local Change Logs Using ODSM

You can view change logs by using Oracle Directory Services Manager, as follows:

1. From the task selection bar, select **Advanced**.
2. Expand **Change Log** if it is not already expanded. The left panel lists the last 500 changes, beginning with the most recent.
3. Select a change to view its properties.

44.2.1.2 Properties of the Change Log Entry

Table 44-2 lists the properties shown on the Change Log page and the corresponding attributes of the change log entry.

Table 44-2 Properties on the Change Log Page

Property	Change Log Attribute
Change Number	`changenumber`
Operation	`changetype`
TargetDN	`targetdn`
Changes	`changes`
Global Unique Identifier (GUID)	`orclguid`
Parent GUID	`orclparentguid`
Change Retry Count	`orclchangeretrycount`
Modifier's Name	`modifiersname`
Operation Time	`operationtime`
Server Name	`servername`

44.3 Overview of Managing and Monitoring Replication Using the Command Line

Understand how to manage and monitor replication using ldapmodify, ldapsearch, remtool and using command line.

This section contains the following topics:

- Enabling and Disabling Change Log Generation Using the Command Line
- Overview of Viewing Change Logs Using ldapsearch
- Overview of Configuring Attributes of the Replica Subentry Using ldapmodify
- Specifying Pilot Mode for a Replica by Using remtool
- Overview of Configuring Replication Agreement Attributes by Using ldapmodify
- Overview of Modifying Replica Naming Context Object Parameters Using ldapmodify
- Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify

- Overview of Monitoring Conflict Resolution Messages Using the Command Line
- Managing the Human Intervention Queue
- Monitoring Replication Progress in a Directory Replication Group Using remtool -pthput
- About Viewing Queue Statistics and Verifying Replication Using remtool
- Managing the Number of Entries the Human Intervention Queue Tools Can Process
- Configuring Replication Filtering Using the `orclEntryExclusionFilter` Attribute

44.3.1 Enabling and Disabling Change Log Generation Using the Command Line

You can enable and disable change log generation by using `ldapmodify` to change the value of `orclgeneratechangelog`, which is an instance-specific attribute. You enable change log generation by setting the value to `1` and disable it by setting the value to `0`.

The command is:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

The LDIF file for changing the value of the `orclgeneratechangelog` attribute in the instance-specific entry to `1` looks like this:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclgeneratechangelog
orclgeneratechangelog: 1
```

44.3.2 Overview of Viewing Change Logs Using ldapsearch

You can view the Change Logs using `ldapsearch`.

You can also view the important attributes in the change log from Table 44-3.

This section includes the following topics:

- Viewing Change Logs Using ldapsearch
- Important Attributes in the Change Log

44.3.2.1 Viewing Change Logs Using ldapsearch

To view change logs from the command line, you use `ldapsearch`. Specify in the search filter the change number or range of change numbers of the change logs you want to view. If you want to view change logs that have been transported from a supplier to the local host, also specify the replica ID of the supplier in the search filter.

Consider the following examples:

- To view a range of change logs that have been transported from the supplier to the local node, type:

  ```
  ldapsearch -D cn=orcladmin -p port -q -b "cn=changelog" -s one \
     "(&(objectclass=changeLogEntry)(servername=SUPPLIER_REPLICAID)\
     (changeNumber>=FROMCHGNO)(changeNumber<=TOCHGNO))"
  ```

- To view a single change log that has been transported from the supplier to the local node, type:

```
ldapsearch -D cn=orcladmin -p port -q -b "cn=changelog" -s one \
   "(&(objectclass=changeLogEntry)(servername=SUPPLIER_REPLICAID)\
   (changeNumber=CHGNO))"
```

- To view a range change logs that have been generated at the local node, type:

```
ldapsearch -D cn=orcladmin -p port -q -b "cn=changelog" -s one \
 "(&(objectclass=changeLogEntry)(changeNumber>=FROMCHGNO)(changeNumber<=TOCHGNO))"
```

- To view a single change log that has been generated at the local node, type:

```
ldapsearch -D cn=orcladmin -p port -q -b "cn=changelog" -s one \
   "(&(objectclass=changeLogEntry)(changeNumber=CHGNO))"
```

Some lines in the output might contain the string `<@! !@>` as a separator.

44.3.2.2 Important Attributes in the Change Log

Table 44-3 lists the important attributes in the change log.

Table 44-3 Important Attributes in the Change Log

Attribute	Description
`changenumber`	Change Number
`changetype`	Operation
`targetdn`	Target DN
`changes`	Changes
`orclguid`	Global Unique Identifier (GUID)
`orclparentguid`	Parent GUID
`orclchangeretrycount`	Change Retry Count
`modifiersname`	Modifier's Name
`operationtime`	Operation Time
`servername`	Server Name
`changeloginfo`	Additional change log information, such as the value of the client IP address For example: `changeloginfo=clientip=::ffff:10.229.116.104`

44.3.3 Overview of Configuring Attributes of the Replica Subentry Using ldapmodify

You can configure replica subentry attributes using `ldapmodify`.

Table 44-4 lists the attributes that can be modified using ldapmodify.

This section includes the following topics:

- Configuring Attributes of the Replica Subentry Using ldapmodify
- Replica Subentry Attributes

44.3.3.1 Configuring Attributes of the Replica Subentry Using ldapmodify

The replica subentry has the DN

```
orclreplicaid=Replica _ID,cn=replication configuration
```

The command line syntax to modify replica subentry attributes using ldapmodify is:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

To set `orclreplicastate` to zero, use the following LDIF file:

```
dn: orcclreplicaid=Replica _ID,cn=replication configuration
changetype: modify
replace: orclreplicastate
orclreplicastate: 0
```

44.3.3.2 Replica Subentry Attributes

Table 44-4 lists the attributes of the replica subentry that you can modify with `ldapmodify`.

Table 44-4 Replica Subentry Attributes

Description	Configuration Attribute
Replica ID	`orclreplicaid`
Replica Primary URI	`orclreplicauri`
Replica Secondary URI	`orclreplicasecondaryuri`
Replica State	`orclreplicastate`
Replica Type	`orclreplicatype`

See Also:

- Table 43-1
- LDAP Replica States

Table A-2 describes the values of `orclreplicastate` in detail. You can set 0, 1 the values, 2, 6, or 8. The other `orclreplicastate` values listed in Table A-2 are read-only values set by the replication server during bootstrap.

44.3.4 Specifying Pilot Mode for a Replica by Using remtool

Before you deploy a replica as part of your enterprise, you might want to test it in pilot mode.

Use the `remtool` command to begin and end pilot mode. The syntax is:

```
remtool -pilotreplica begin -bind hostname:ldap_port
```

```
remtool -pilotreplica end -bind hostname:ldap_port [-bkup file_name]
```

When you run remtool -pilotreplica begin:

- `orclreplicatype` is set to `2` (pilot)
- `orclpilotmode` is set to `1`
- `pilotstarttime` is set to current time.

When you run remtool -pilotreplica end

- `orclpilotmode` is set to `0`

Do not attempt to modify these attributes directly with `ldapmodify`.

> **See Also:**
>
> The `remtool` command reference in *Reference for Oracle Identity Management* for more information about the `-pilotreplica` option to `remtool`.

44.3.5 Overview of Configuring Replication Agreement Attributes by Using ldapmodify

You can configure replication agreement attributes using `ldapmodify`.

This section includes the following topics:

- Replication Agreement Options
- Configuring Replication Agreement Attributes Using ldapmodify

44.3.5.1 Replication Agreement Options

Table 44-5 lists the replication agreement attributes that you can modify by using `ldapmodify`. See Table 43-2 for more information.

Table 44-5 Replication Agreement Options

Description	Configuration Attribute	Default Value
Replication frequency	`orclupdateschedule`	60 (seconds)
HIQ Schedule	`orclhiqschedule`	600 (seconds)
Whether the connections from the directory replication server to the directory server are kept active or established every time changelog processing is done	`orclldapconnkeepalive`	1

> **See Also:**
>
> Table 43-2.

44.3.5.2 Configuring Replication Agreement Attributes Using ldapmodify

The replication agreement has the DN:

`orclagreementid=`*`Agreement_ID`*`,orclreplicaid=`*`Replica_ID`*`,cn=replication configuration`

> **Note:**
>
> Always inactivate replication before you delete or modify a replication agreement.

The following LDIF file changes the human intervention queue schedule by changing the value of the `orclHIQSchedule` attribute in the replication agreement to 900 minutes:

```
dn: orclagreementid=Agreement_ID,orclreplicaid=Replica_ID,cn=replication configuration
changetype: modify
replace: orclhiqschedule
orclhiqschedule: 900
```

44.3.6 Overview of Modifying Replica Naming Context Object Parameters Using ldapmodify

It is possible to change the replication scope from the command line.

To change the replication scope, you must create or modify the naming context object entries under the replication naming context container entry.

This section includes the following topics:

- Modifying Replica Naming Context Object Parameters Using ldapmodify
- Adding a Naming Context Object for an LDAP-Based Replica
- Deleting a Naming Context Object
- Modifying the orclIncludedNamingContexts Attribute for a Replica Naming Context Object
- Modifying the orclExcludedNamingContexts Attribute for a Replica Naming Context Object
- Modifying the orclExcludedAttributes Attribute for a Replica Naming Context Object

> **See Also:**
>
> - Table 43-2
> - Replication Naming Context Container Entry
> - Understanding Replication Naming Context Object Entry

44.3.6.1 Modifying Replica Naming Context Object Parameters Using ldapmodify

To change the replication scope, use a command line such as:

```
ldapadd -p port_number -h host -f file.ldif
```

with an LDIF file to set the scope for an agreement.

For example, use the following LDIF file to set the replication scope to `cn=oraclecontext`:

```
dn: cn=includednamingcontext000001,cn=replication
 namecontext,orclagreementid=agreementid,orclreplicaid=replicaid,cn=replication
 configurationobjectclass: orclreplnamectxconfig
```

```
orclincludednamingcontexts: cn=oraclecontext
cn: includednamingcontext000001
```

Use the following file to exclude EMEA and APAC groups and exclude the attributes userpassword and authpassword from the replication scope

```
dn: cn=includednamingcontext000002,cn=replication
 namecontext,orclagreementid=agreementid,orclreplicaid=replicaid,cn=replication
 configuration
objectclass=orclreplnamectxconfig
orclincludednamingcontexts: dc=com
orclexcludednamingcontexts: cn=groups,l=emea,dc=xyz,dc=com
orclexcludednamingcontexts: cn=groups,l=apac,dc=xyz,dc=com
orclExcludedAttributes: userpassword
orclExcludedAttributes: authpassword
cn: includednamingcontext000002
```

Replica naming context object parameters are listed in Oracle Directory Replication Schema Elementsin *Reference for Oracle Identity Management*.

> **Note:**
>
> The replication server reads naming context objects from the supplier replica.

44.3.6.2 Adding a Naming Context Object for an LDAP-Based Replica

You can create a naming context object for an LDAP-based replica, that does the following:

- Replicates the naming context `ou=Americas,cn=mycompany`
- Excludes from replication the naming context `cn=customer profile, ou=Americas,cn=mycompany`
- Excludes from replication the attribute `userpassword`

Follow the steps below to add a naming content object:

1. Edit the example file `mod.ldif` as follows:

   ```
   dn: cn=naming_context_identifier, cn=replication  namecontext,
    orclagreementid=replication_agreement_identifier,
    orclreplicaid=supplier_replica_identifier,cn=replication configuration
   orclincludednamingcontexts: ou=Americas,cn=mycompany
   orclexcludednamingcontexts: cn=customer profile, ou=Americas, cn=mycompany
   orclexcludedattributes: userpassword
   objectclass: top
   objectclass: orclreplnamectxconfig
   ```

2. Use ldapadd to add the partial replication naming context object to the supplier.

   ```
   ldapadd -D "cn=orcladmin" -q -h supplier_host \
           -p port_number -f mod.ldif
   ```

44.3.6.3 Deleting a Naming Context Object

To delete the naming context object created in Adding a Naming Context Object for an LDAP-Based Replica, type:

```
ldapdelete -D "cn=orcladmin" -q \
           -h supplier_host -p supplier_host_port_number \
           "cn=naming_context_identifier, cn=replication namecontext, \
            orclagreementid=replication_agreement_identifier, \
            orclreplicaid=supplier_replica_identifier, \
            cn=replication configuration"
```

44.3.6.4 Modifying the orclIncludedNamingContexts Attribute for a Replica Naming Context Object

The directory replication server uses the `orclIncludedNamingcontexts` attribute value of the replica naming context object to specify the top-level subtree included in partial replication.

In this example, the included naming context is set to `c=us`, which means that `c=us` is to be included in partial replication.

1. Edit the example file `mod.ldif` as follows:

```
DN:cn=naming_context_identifier,cn=replication namecontext,
 orclagreementid=replication_agreement_identifier,
 orclreplicaid=supplier_replica_identifier,cn=replication configuration
Changetype:modify
Replace: orclIncludedNamingcontexts
orclIncludedNamingcontexts: c=us
```

2. Use ldapmodify to update the replication agreement orclupdateschedule attribute.

```
ldapmodify -D "cn=orcladmin" -q -h supplier_host -p port -f mod.ldif
```

3. Restart the directory replication server.

44.3.6.5 Modifying the orclExcludedNamingContexts Attribute for a Replica Naming Context Object

The directory replication server uses the `orclExcludedNamingcontexts` attribute value of the replica naming context object to specify the top-level subtrees excluded from partial replication.

In this example, the excluded naming contexts are set to `ou=Europe,c=us` and `ou=Americas,c=us`, which means that these two naming contexts are to be excluded from partial replication.

1. Edit the example file `mod.ldif` as follows:

```
DN:cn=naming_context_identifier,
 cn=replication namecontext,
 orclagreementid=replication_agreement_identifier,
 orclreplicaid=supplier_replica_identifier,cn=replication configuration
Changetype:modify
Replace: orclExcludedNamingcontexts
orclExcludedNamingcontexts: ou=Europe, c=us
orclExcludedNamingcontexts: ou=Americas, c=us
```

2. Use ldapmodify to update the replication agreement orclupdateschedule attribute.

```
ldapmodify -D "cn=orcladmin" -q -h supplier_host -p port -f mod.ldif
```

3. Restart the directory replication server.

> **Note:**
>
> A subtree specified in the `orclexcludednamingcontexts` attribute must also be a subtree of the specified `includednamingcontext` of the same replica naming context object.

44.3.6.6 Modifying the orclExcludedAttributes Attribute for a Replica Naming Context Object

You can specify that certain changes made to the included naming context be excluded, at attribute level, from partial replication. To determine which attributes are to be excluded, the directory replication server uses the value of the orclExcludedAttributes attribute of the replica naming context object.

In this example, the `telephonenumber` and `title` attributes of the naming context specified in the orclincludednamingcontexts attribute are excluded from replication.

1. Edit the example file `mod.ldif` as follows:

```
DN:cn=naming_context_identifier,
 cn=replication namecontext,
 orclagreementid=replication_agreement_identifier,
 orclreplicaid=supplier_replica_identifier,cn=replication configuration
Changetype:modify
Replace: orclExcludedAttributes
orclExcludedAttributes: telephonenumber
orclExcludedAttributes: title
```

2. Use ldapmodify to update the replication agreement orclupdateschedule attribute.

```
ldapmodify -D "cn=orcladmin" -q -h my_host -p port -f mod.ldif
```

3. Restart the directory replication server.

44.3.7 Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify

You can configure attributes f the replication configuration set using `ldapmodify`.

This section includes the following topics:

- About Configuring Attributes of the Replication Configuration Set Using ldapmodify
- Replication Configuration Attributes and Debug Levels

44.3.7.1 About Configuring Attributes of the Replication Configuration Set Using ldapmodify

The replication configuration set has the DN:

```
cn=configset0,cn=osdrepld,cn=subconfigsubentry
```

For example, the following LDIF file enables SASL for replication binds by adding and setting the `orclreplusesasl;digest-md5` attribute in the replication configuration set:

```
dn: cn=configset0,cn=osdrepld,cn=subconfigsubentry
changetype: modify
add: orclreplusesasl;digest-md5
orclreplusesasl;digest-md5: 1
```

See Managing the Number of Entries the Human Intervention Queue Tools Can Process for information about changing `orclsizelimit`.

> **See Also:**
>
> Table 43-4.

44.3.7.2 Replication Configuration Attributes and Debug Levels

Table 44-6 lists the replication configuration set attributes that you can modify with `ldapmodify`.

Table 44-6 Replication Configuration Attributes

Description	Configuration Attribute	Default Value
Change Retry Count	`orclchangeretrycount`	10
Maximum Number of Workers	`orclreplmaxworkers`	20
Autotune Replication (Restart server after changing.)	`orclreplautotune`	1
Number of Apply Threads Per Supplier	`orclthreadspersupplier;appll y`	5
Number of Transport Threads per Supplier	`orclthreadspersupplier;trans port`	1
Maximum Number of Entries to Process per Replication Cycle	`orclsizelimit`	1000
Automatically Resolve Replication Conflicts	`orclconflresolution`	1
Generate Stack Dump (Restart server after changing.)	`orclsdumpflag`	
SASL for Replication Bind	`orclreplusesasl;digest-md5`	none
Maximum Log File Size (MB)	`orclmaxlogfilesize`	1
Maximum number of log files to keep in rotation	`orclmaxlogfiles`	100
DebugLevel	`orcldebuglevel`	0
Replication Status	`orclreplicationstate`	
Activate/Inactivate	`orclactivatereplication`	0

The attribute `orcldebuglevel` can be set to any combination of the values shown in Table 44-7. The values are additive. No restart is required.

Table 44-7 Replication Debug Levels

Debug Level	Value of orcldebuglevel
Replication Process Trace	4194304
Replication Performance Log	2097152
Trace Function Calls	8388608
Heavy Trace Debugging	16777216

44.3.8 Overview of Monitoring Conflict Resolution Messages Using the Command Line

You can monitor conflict resolution messages using the command line.

Table 44-8 lists the change types and resolution for each conflict types.

This section contains the following topics:

- Monitoring Conflict Resolution Messages Using the Command Line
- Conflict Resolution Messages

44.3.8.1 Monitoring Conflict Resolution Messages Using the Command Line

If a conflict has been written into the log, then it means that the system is not able to resolve it by following its resolution procedure. To avoid further replication change conflicts arising from earlier unapplied changes, it is important to monitor the logs regularly.

To monitor replication change conflicts, examine the contents of the replication log. You can distinguish between messages by their respective timestamps.

Conflict resolution messages are logged in the file *$DOMAIN_HOME*/servers/OID/logs/*componentName*/oidrepld00-XXXX.log where *XXXX* is a number from 0000 to orclmaxlogfiles configured.

Each message includes the conflict reason, change number, supplier node, change type, target DN, and result. Here are some examples:

This is a conflict resolution message where the conflict was automatically resolved by replication server:

```
[2008-10-09T09:57:31-07:00] [OID] [NOTIFICATION:16] [] [OIDREPLD] [host: stacu14]
[pid: 4280] [tid: 3] Worker(Transport)::[[************ Conflict Resolution
Message ************
Conflict reason: Attempted to add an existing entry.
Change number: 3696.
Supplier: stacu14_lm5.
Change type: Add.
Target DN: dc=org.
Result: Dropped a newer change entry.
]]
```

This is an unresolved conflict which was moved to Human intervention queue:

```
[2008-10-09T10:03:28-07:00] [OID] [NOTIFICATION:16] [] [OIDREPLD] [host: stacu14]
[pid: 4653] [tid: 13] Worker(Transport)::[[************ Conflict Resolution
Message ************
```

```
Conflict reason: Attempted to delete a non-existent entry.
Change number: 3698.
Supplier: stacu14_lm5.
Change type: Delete.
Target DN: dc=imc,dc=org.
Result: Change moved to low priority queue after failing on 10th retry.
]]
```

44.3.8.2 Conflict Resolution Messages

Table 44-8 lists the conflict reasons, change types, and results.

Table 44-8 Conflict Resolution Messages

Conflict Reason	Change Type	Result
Attempted to add an existing entry	Add	Dropped a newer change entry. Change moved to low priority queue after failing on Nth retry Deleted duplicated target entry which was created later than the change entry. Apply the change entry again Dropped a newer change entry Dropped change entry since its guid is greater than target entry guid Dropped change entry since its guid is same as target entry i.e change and target entry are identical Deleted duplicated target entry which has greater guid than the change entry
Parent entry does not exist	Add	Change moved to low priority queue after failing on Nth retry
Internal Error occurred	Add Delete Modify moddn	Change moved to low priority queue after failing on Nth retry
Attempted to modify a non-existent entry	Modify	Change moved to low priority queue after failing on Nth retry
Attempted to delete a non-existent entry	Delete	Change moved to low priority queue after failing on Nth retry
Attempted to delete a non leaf entry	Delete	Change moved to low priority queue after failing on Nth retry
Attempted to move a non-existent entry	moddn	Change moved to low priority queue after failing on Nth retry
Attempted to move to an existent entry	moddn	Change moved to low priority queue after failing on Nth retry
Synchronization of moddn for another non-existent entry is going on	moddn	Deleted existent entry which was created later than the entry trying to move. Move of entry is tried again
Attempted to move to an existent entry which is newer than the source entry	moddn	Move of entry is tried again

Table 44-8 (Cont.) Conflict Resolution Messages

Conflict Reason	Change Type	Result
Attempted to move to an existent entry which is older than the entry trying to move	moddn	Deleted Source entry and dropped the change
Attempted to move to an existing entry which was created at the same time and had the same guid	moddn	Deleted Source entry and dropped the change
Attempted to move to an existing entry which has greater guid than the entry trying to move	moddn	Deleted duplicated target entry and re-applied the change
Attempted to move to an existing entry which has lower guid then entry trying to move	moddn	Deleted Source entry and dropped the change

44.3.9 Managing the Human Intervention Queue

When a replication conflict arises, the Oracle Internet Directory replication server places the change in the retry queue and tries to apply it from there for a specified number of times. If it fails after the specified number of retries, the replication server puts the change in the human intervention queue. From there, the replication server repeats the change application process at less frequent intervals while awaiting your action.

The human intervention queue tools, `ManageHiq.retry` and `ManageHiq.purge`, enable you to move changes from the human intervention queue to the retry queue or the purge queue, respectively. After you move the change to the purge queue, there are no further attempts to re-apply the changelog entry. To address changes in the human intervention queue, follow these general steps:

1. Examine the change in the human intervention queue.
2. Reconcile the conflicting changes using the Compare and Reconcile Tool (see Overview of Comparing and Reconciling Inconsistent Data Using oidcmprec).
3. Either place the change back into the retry queue using `ManageHiq.retry` or into the purge queue using `ManageHiq.purge`.

> **See Also:**
>
> The Oracle Internet Directory Replication Management Tools in *Reference for Oracle Identity Management* for instructions on how to use the Human Intervention Queue tools

44.3.10 Monitoring Replication Progress in a Directory Replication Group Using remtool -pthput

The Replication Environment Management Tool, `remtool`, enables you to monitor replication progress in a directory replication group.

When you invoke `remtool` with the `-pthput` option, the tool binds to the specified node and collects information about all the nodes in the directory replication group. It displays this information at intervals of specified duration.

The syntax is:

```
remtool -pthput [-bind hostname:port] -interval time_in_seconds [-file filename]
```

The `bind` argument is optional. If you do not provide it, `remtool` prompt you for `hostname`, `port`. You are prompted for the replication dn password.

The `interval` parameter is an optional parameter. Provide its value in seconds. Its default value is 60 seconds. After each interval the tool displays the following information for every supplier and consumer node in the directory replication group:

- The changes that were queued in the last interval, `q_chgs`
- Last applied change number, `LA_Chg#`
- Changes applied in the last interval, `Appl_chgs`
- Average throughput for the latest three intervals, `Avg_thput`

If you specify a `file` parameter, the output shown on the command line is logged to that file. Otherwise, the output is logged to a file name based on the timestamp.

Example Output

```
--------------------------------------------------------------------------------
Directory Replication Group (DRG) details :
--- ------------------ ---------------------- ---------------------- -----
Sl  Replicaid          Directory Information   Supplier Information    Repl.
No.                                                                   Type
--- ------------------ ---------------------- ---------------------- -----
001 adc2101322_nldap32 adc2101322.us.example.com:3070 adc2101322_nldap3      RW
002 adc2101322_nldap3  adc2101322.us.example.com:3061 adc2101322_nldap32     RW
--- ------------------ ---------------------- ---------------------- -----
Queue Statistics :
------------ ------------- ----- ------- ------ ------ ------ --------- ---------
--------------------------------
     Supplier            Consumer           Queued      Last Applied     Applied
Average
                                            Changes     Change Number    Changes
Throughput
                                                                                 Of
3 intervals
------------ ------------- ----- ------- ------ ------ ------ --------- ---------
------------------------------
adc2101322_nldap3    adc2101322_nldap32        0          134369
0            0
adc2101322_nldap32   adc2101322_nldap3         0           47342
0            0
------------ ------------- ----- ------- ------ ------ ------ --------- ---------
--------------------------
```

```
adc2101322_nldap3    adc2101322_nldap32     2286          136214          1845
615
adc2101322_nldap32   adc2101322_nldap3         0           47342
0             0
------------ ------------ ----- ------- ------ ------ ------ --------- ---------
----------------------
adc2101322_nldap3    adc2101322_nldap32     1608          137886          1672
1172
adc2101322_nldap32   adc2101322_nldap3         0           47342
0             0
------------ ------------ ----- ------- ------ ------ ------ --------- ---------
-----------------------
adc2101322_nldap3    adc2101322_nldap32      189          140302          2416
1977
adc2101322_nldap32   adc2101322_nldap3         0           47342
0             0
------------ ------------ ----- ------- ------ ------ ------ --------- ---------
----------------------
```

44.3.11 About Viewing Queue Statistics and Verifying Replication Using remtool

The Replication Environment Management Tool, `remtool`, enables you to monitor the health of the replication process.

You can run `remtool` periodically to ensure that your replication processes are performing properly. The `remtool` command has options that display queue statistics and verify replication.

The `remtool` options for monitoring an LDAP-based replication agreement are `-pdipqstat` and `-pverify`. Their syntax is as follows:

```
remtool -pdispqstat [-v] [-bind hostname:port_number]

remtool -pverify [-v] [-bind hostname:port_number] [-hiqmax hiqmax] [-tbtmax tbtmax]
```

All of these commands prompt for the `replication_dn_password`.

First run `remtool` with the `-pdispqstat` option. It shows the queue statistics of the DRG. Check to see if the number of Human Intervention Queue (HIQ) entries and change logs to be transported (Logs TBP) are higher than usual. If so, that means replication is running more slowly than it should. Run `remtool` with the `-pverify` option to verify your replication configurations.

If `remtool` with the `-pverify` option reports test failure, check the report that it generates and follow the suggestion in the report to fix the specific failures.

> **See Also:**
>
> The `remtool`command reference in *Reference for Oracle Identity Management*

44.3.12 Managing the Number of Entries the Human Intervention Queue Tools Can Process

To increase the number of changelogs processed at a time, you must set `orclsizelimit` in the replication configuration set and `orclsizelimit` in the server instance where replication is running to the same value, a value greater than 1000. Use `ldapmodify` to change both of them.

In each case, type:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

To set `orclsizelimit` in the replication configuration set to 5000, use an LDIF file such as:

```
dn: cn=configset0,cn=osdrepld,cn=subconfigsubentry
changetype: modify
replace: orclsizelimit
orclsizelimit: 5000
```

To set `orclsizelimit` in the server instance to 5000, use an LDIF file such as:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orclsizelimit
orclsizelimit: 5000
```

Setting the Oracle Internet Directory server instance parameter `orclsizelimit` very high impacts server performance, because `orclsizelimit` also controls the maximum number of entries to be returned by a search.

See Also:

- Attributes of the Instance-Specific Configuration Entry
- Understanding Replication Configuration Set
- Overview of Configuring Attributes of the Replication Configuration Set by Using ldapmodify

44.3.13 Configuring Replication Filtering Using the `orclEntryExclusionFilter` Attribute

Oracle Internet Directory server and the directory replication server allow the exclusion of specific entries based on an LDAP filter configured in the `orclEntryExclusionFilter` attribute in the replication agreement.

The `orclEntryExclusionFilter` attribute is an optional, single-valued attribute that specifies a valid LDAP filter string enclosed in parenthesis.

> **Note:**
>
> The LDAP filter string in `orclEntryExclusionFilter` attribute does not support non-catalogued attributes and collective attributes. If specified the filter is not applied.

Replication filtering using the `orclEntryExclusionFilter` attribute applies only to one-way LDAP replication, and other replication functionality remains unchanged.

To configure replication filtering using the `orclEntryExclusionFilter` attribute:

1. Use `ldapsearch` on `"cn=replication configuration"` to find the `dn` of the replication agreement entry.
2. Use `ldapmodify` to add the `orclEntryExclusionFilter` attribute to the replication agreement entry from Step 1. In the following example, the `orclEntryExclusionFilter` attribute specifies the LDAP filter as `(sn=smith)`:

```
dn: orclagreementid=000001,orclreplicaid=supplier,cn=replication configuration
changetype: modify
add: orclentryexclusionfilter
orclentryexclusionfilter: (sn=smith)
```

3. Restart the replication server for the LDAP filter to take effect.

Subsequent changes for all entries matching the LDAP filter `(sn=smith)` at the supplier are not replicated to the consumer.

44.4 Overview of Comparing and Reconciling Inconsistent Data Using oidcmprec

Understand about the `oidcmprec` tool and some examples of `oidcmprec` usage.

> **Note:**
>
> The Compare and Reconcile Tool, `oidcmprec` does not support One way or Two way Authentication and works only on the No-Authentication mode.

This section contains the following topics:

- Comparing and Reconciling Inconsistent Data Using oidcmprec
- Conflict Scenarios
- Operations Supported by oidcmprec
- Output from oidcmprec
- How oidcmprec Works
- Source and Destination Directories Setup
- DIT for the oidcmprec Operation
- Attributes Selection for the Operation
- Control of Change Log Generation

- oidcmprec Command-Line Arguments Specification in a Text or XML Parameter File
- Directory Schema Inclusion in oidcmprec
- Override of Predefined Conflict Resolution Rules
- User-Defined Compare and Reconcile Operation
- Known Limitations of the oidcmprec Tool

> **See Also:**
>
> The oidcmprec command reference in *Reference for Oracle Identity Management* for the complete syntax for `oidcmprec`, including operations, conflict scenarios, and conflict resolution rules.

44.4.1 Comparing and Reconciling Inconsistent Data Using oidcmprec

When the directory replication server encounters inconsistent data, you can use the Oracle Internet Directory Comparison and Reconciliation Tool to synchronize the entries on the consumer with those on the supplier.

Perform the following general steps:

1. Set the supplier and the consumer to read-only mode. Use one of the procedures in Changing the Server Mode.
2. Ensure that the supplier and the consumer are in a tranquil state—that is, that neither is supplying or applying changes. If they are not in a tranquil state, then wait until they have finished updating.
3. Identify the inconsistent entries or subtree on the consumer.
4. Use the Oracle Internet Directory Comparison and Reconciliation Tool to fix the inconsistent entries or subtree on the consumer.
5. Set the participating supplier and consumer back to read/write mode.

> **See Also:**
>
> The `oidcmprec`command-line tool reference in *Reference for Oracle Identity Management* for syntax and an explanation of how Oracle Internet Directory Comparison and Reconciliation Tool works.

The compare and reconcile tool `oidcmprec` enables you to compare two directories. It detects and resolves conflicts. Of the two directories, one directory is considered to be the source directory or "source of truth." The other directory is the destination directory that must be synchronized with the source directory. The directories to be compared can be directories that are not part of any replication group, part of the same replication group, or part of different replication groups.

44.4.2 Conflict Scenarios

The `oidcmprec` tool can detect and resolve various conflict scenarios:

The `oidcmprec` tool can detect and resolve the following conflict scenarios:

- Entry only in source directory (entos)
- Entry only in destination directory (`entod`)
- Attribute only in source directory (`atros`)
- Attribute only in destination directory (`atrod`)
- Single-valued attribute differs (`svatrdif`)
- Multi-valued attribute differs (`mvatrdif`)
- Entry DN differs (`dndif`)

  The `dndif` scenario can occur in a replication environment when a `modrdn` or `moddn` operation performed in one node is not replicated to another node. As a result, the entry has the same `orclguid` but different DNs on the two nodes. The tool uses the `orclguid` attribute to detect this conflict.

The `oidcmprec` tool can also detect and resolve the following schema conflict scenarios:

- Object class definition exists only in source directory (`odefos`)
- Object class definition exists only in destination directory (`odefod`)
- Object class definition different in source and destination directory (`odefdif`)
- Attribute definition exists only in source directory (`adefos`)
- Attribute definition exists only in destination directory (`adefod`)
- Attribute definition different in source and destination directory (`adefdif`)

44.4.3 Operations Supported by oidcmprec

The tool supports five operation. Each operation compares entries, detects conflicts, and optionally resolves them. The operations differ regarding how they resolve conflicts.

The operations are as follows:

- Compare operation: compares two directories and stores the changes as LDIF records in a file. The LDIF file can be applied to the destination directory to make it identical to the source directory. Only the data in the source directory is considered valid.
- Reconcile operation: compares two directories and applies necessary changes at the destination directory to make it identical to the source directory. All the changes made to the directory are stored as LDIF records in a file. Only the data in the source directory is considered valid.
- Merge or two-way reconcile operation: compares two directories and applies necessary changes at the source or destination directory to make them identical. The data in both directories is considered valid. For example, when the tool detects that an entry exists only in the destination directory, the tool adds it to the source. This operation also records all the changes applied to the directory as LDIF records in a file.
- Merge dry run operation: compares two directories, similar to the merge operation, but does not apply the changes in the directory. Instead, it stores the changes as LDIF records in a file. The changes to be applied to the source directory and to destination directory are stored in two different files.
- User defined compare and reconcile operation: uses conflict resolution rules that you choose for each conflict scenario. See *Reference for Oracle Identity Management* for a list of conflict resolution rules you can use for each conflict scenario.

44.4.4 Output from oidcmprec

The `oidcmprec` tool normally generates several output files. You can use options to `oidcmprec` to suppress generation of any of the files.

The files and their corresponding options are:

- *`filename`*`.rpt`: contains the DNs of all entries compared and the compare result. Use `logrpt=false` to suppress generation of this file.
- *`filename`*`.s2d.ldif`: contains all changes applied to the destination directory or stored for later application to the destination directory. The name is an abbreviation for source directory to destination directory. Use `logs2d=false` to suppress generation of this file.
- *`filename`*`.d2s.ldif`: contains all changes applied to the source directory or stored for later application to the source directory. The name is an abbreviation for destination directory to source directory. Use `logd2s=false` to suppress generation of this file.
- *`filename`*`.eos.rpt`: lists DNs of entries that exist only in the source directory. It also lists name of attributes and object classes that are defined only in the source directory, if the schema is included for the operation.The name is an abbreviation for entries available only in source directory. Use `logeos=false` to suppress generation of this file.
- *`filename`*`.eod.rpt`: lists DNs of entries that exist only in the destination directory. It also lists name of attributes and object classes that are defined only in the destination directory, if the schema is included for the operation. The name is an abbreviation for entries available only in destination directory. Use `logeod=false` to suppress generation of this file.
- *`filename`*`.dif.rpt`: lists DNs of all entire that differ along with names of attribute that differ. It also lists name of attributes and object classes whose definitions differ, if the schema is included for the operation. This file is known as the dif file. Use `logdif=false` to suppress generation of this file.
- *`filename`*`.err`: contains all error messages. This file is known as the err file. Use `logerr=false` to suppress generation of this file.

The tool can dump the total number of entries loaded by the tool in memory and the number of entries in each of `oidcmprec`'s various queues. The entry counts are logged in the file `oidcmprec.log`. Use the `qlogfreq=`*`frequency`* argument to specify how frequently `oidcmprec` logs this information. Possible *`frequency`* values are from 1 to 5000. The lower the value, the shorter the interval. For frequent entry counts, use a value between 5 and 10.

44.4.5 How oidcmprec Works

Using the replication DN and replication DN password as credentials, the tool performs three tasks. First it loads schema information into memory. Next, it collects the entries to be compared and it compares them, attribute by attribute. The tool uses schema information to determine the compare rule to be used for each attribute. Then, based on the compare result, the tool takes the necessary actions.

These operations are performed by different threads:

- The DN thread is responsible for collecting entries to be compared. While collecting entries, it does not fetch the entire tree at once. It first fetches the base entry and processes it. Then it fetches the immediate children of the base entry and processes them, and then their immediate children, and so on. The collected entries are passed on to worker threads. You can control the number of DN threads using the `dnthreads` argument.

- The worker thread is responsible for comparing entries attribute by attribute and applying conflict resolution rules. The worker thread then passes on the entry to the log writer thread. You can control the number of worker threads by using the `threads` argument. The total number of worker threads and DN threads cannot exceed a maximum value equal to 6 * (Number of CPUs) - 2. If you specify more than that, the tool adjusts the number of worker and DN threads so as not to exceed the maximum.
- The log writer thread is responsible for writing the contents to all seven output files listed in Output from oidcmprec. There is only one log writer thread. You cannot increase the number.

These threads are spawned, monitored and terminated by the main thread. The main thread processes command line arguments and parameter files and spawns the other threads. As soon as the main thread detects that all operations are complete, it terminates all threads and cleans up all connections.

Each thread establishes an LDAP connection to the source directory and to the destination directory. These connections remain open until all operations are completed by the thread. If a connection is closed for any reason, the tool tries to reestablish connection if the `continueOnError` argument is set to TRUE. If the tool can reestablish the connection, it continues operation.

> **Note:**
>
> Use the `continueOnError` argument to specify whether the tool should continue processing on error. This argument can be set to `TRUE` or `FALSE`. By default it is set to `TRUE`.

44.4.6 Source and Destination Directories Setup

You use the `source` and `destination` options to set the source and destination directories.

You can set source and destination options as shown below:

```
oidcmprec source=staqj13:3060 destination=staqj:3070 base="'" \
          scope=subtree file=temp operation=compare
```

The tool prompts for passwords if they are not provided on the command line:

```
Enter replication DN password of the source directory     :
Enter replication DN password of the destination directory :
```

44.4.7 DIT for the oidcmprec Operation

Use the `base`, `dns2Exclude`, and `scope` options to choose the area to be compared and reconciled.

This example compares the entire directory except `c=us,dc=mycom,dc=com` and `c=uk,dc=mycom,dc=com`:

```
oidcmprec base="'" \
          dns2exclude="'c=us,dc=mycom,dc=com' 'c=uk,dc=mycom,dc=com'" \
          operation=compare scope=subtree \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          threads=5 dnthreads=2 file=cmpres
```

This example compares the naming contexts `dc=com` and `dc=org` except for the trees `c=us,dc=mycom,dc=com` and `c=uk,dc=myorg,dc=org`.

```
oidcmprec base="'dc=com' 'dc=org'" \
          dns2exclude="'c=us,dc=mycom,dc=com' 'c=uk,dc=myorg,dc=org'" \
          operation=compare scope=subtree \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          threads=5 dnthreads=2 file=cmpres
```

44.4.8 Attributes Selection for the Operation

By default, `oidcmprec` compares all attributes except for the operational attributes `creatorsname`, `createtimestamp`, `modifiersname`, `modifytimestamp`, `orclentrydn`, and `orclnormdn`.

You can control the attributes to be included or excluded for the chosen operation using `excludedAttributes` or `incudedAttributes`, respectively. The `excludedAttributes` and `incudedAttributes` arguments allow limited pattern matching. You can use *`attributename*`* to match all attributes starting with *`attributename`*. You can also use *`attributename;*`* to match all subtypes of *`attributename`*.

This example excludes the `authpassword` attribute with and without subtype, plus the attributes `userpassword` and `category`, in addition to the standard excluded attributes.

```
oidcmprec operation=compare scope=subtree base="'dc=com' 'dc=org'" \
          source=myhost1.mycom.com:3060\
          destination=myhost2.mycom.com:3060 \
          exclattr="authpassword authpassword;* userpassword category" \
          threads=5 dnthreads=2 file=compare
```

This example includes only the attributes `uid`, `cn`, `sn`, `givenname`, and `mail` in the compare operation.

```
oidcmprec operation=compare scope=subtree base="'dc=com'" \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          inclattr="uid cn sn givenname mail" \
          file=compare
```

This example includes all attributes for the compare operation except `orclguid`, `creatorsname`, and `modifiersname`. This example also asks the tool to stop when it encounters any error by setting `continueOnError=false`.

```
oidcmprec operation=compare scope=subtree base="'dc=com'" \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          inclattr="*" exclattr="orclguid creatorsname modifiersname" \
          file=compare contonerr=false
```

44.4.9 Control of Change Log Generation

Change log generation for the changes made by `oidcmprec` depends on the value of the `orcldiprepository` attribute of the root DSE. Change log generation behavior, however, can be controlled by using the `generateChangeLog` argument.

The `generateChangeLog` argument can have the following values:

- `default`: The directory server settings determine whether a change log is generated or not. Change logs are generated if the root entry's `orcldiprepository` attribute is set to `true`. They are not generated if `orcldiprepository` is set to `false`. The same rule applies for both the source and destination directories. `default` is the default value for `gechglog`.
- `true`: Change logs are always generated, irrespective of the settings on the source and destination directories.
- `false`: Change logs are never generated, irrespective of the settings on the source and destination directories.

In the following example, `generateChangeLog false` turns off change log generation:

```
oidcmprec operation=merge scope=subtree base="'dc=com'" \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          inclattr="*" exclattr="orclguid creatorsname modifiersname" \
          file=merge genchglog=false
```

44.4.10 oidcmprec Command-Line Arguments Specification in a Text or XML Parameter File

You can also specify the `oidcmprec` command-line arguments in a parameter file.

You specify a text parameter file using the `parameterFile` option or an XML parameter file using the `xmlparameterFile` option. If you specify an argument both on the command line and in a parameter file, the argument specified on the command line takes precedence over the one specified in the parameter file. For example:

```
oidcmprec paramfile=comp_param threads=4
```

This example uses the following sample text parameter file:

```
##########################################
#Parameter file for compare and reconcile tool
#Creator   : Admin
#Date      : 21-Mar-2012
#File Name : comp_param
##########################################
operation=compare
source=staqj13:3060/ods
destination=staqj13:3070/ods
base='("cn=oraclecontext" "c=uk,dc=example,dc=com" "c=us,dc=example,dc=com")'
verbose=false
force=true
threads=6
dnthreads=2
exclattr=orclguid userpassword authpassword;*
filename=cmp2012Feb01
```

In this example, the tool spawns four worker threads and gives precedence to command-line arguments.

The following XML parameter file is equivalent to the sample text parameter file:

```
  <?xml version="1.0" standalone="yes" ?>
- <input>
  <operation>compare</operation>
- <source>
   <host>staqj13</host>
   <port>3060</port>
```

```
   <binddn>cn=orcladmin</binddn>
   <password>source-password</password>
  </source>
- <destination>
   <host>staqj13</host>
   <port>3070</port>
   <binddn>cn=orcladmin</binddn>
   <password>destination-password</password>
  </destination>
  <base>
   <dn>cn=oraclecontext</dn>
   <dn>c=uk,dc=example,dc=com</dn>
   <dn>c=us,dc=example,dc=com</dn>
  </base>
  <threads>6</threads>
  <dnthreads>2</dnthreads>
  <exclattr>
   <attribute>orclguid</attribute>
   <attribute>userpassword</attribute>
   <attribute>authpassword</attribute>
  <exclattr/>
  <force>true</force>
  <verbose>false</verbose>
  <filename>cmp2012Feb01</filename>
  </input>
```

44.4.11 Directory Schema Inclusion in oidcmprec

You can include the schema in an `oidcmprec` operation by including `cn=subschemasubentry` in the base argument.

For example:

```
oidcmprec operation=merge scope=subtree \
          base="'dc=com' 'cn=subschemasubentry'" \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          inclattr="*" exclattr="orclguid creatorsname modifiersname" \
          file=merge genchglog=false
```

If you include other DNs in addition to the schema, `oidcmprec` performs the operation on the schema first.

44.4.12 Override of Predefined Conflict Resolution Rules

You can override the predefined conflict resolution rules by specifying the conflict name and the rule to use in the command line or parameter file.

Conflict scenarios and conflict resolution rules for each operation are described in the oidcmpreccommand reference in *Reference for Oracle Identity Management*.

The following example changes the conflict resolution rule used for the conflicts `dndif` and `mvatrdif` to `ignore` for the `compare` operation:

```
oidcmprec operation=compare source=host1:3060 destination=host2:3070 \
          base="''" scope=subtree file=temp operation=compare \
          dndif=ignore mvatrdif=ignore
```

44.4.13 User-Defined Compare and Reconcile Operation

The following command line uses the `userdefinedcr` operation:

In addition to the predefined operations `compare`, `reconcile`, `merge`, and `merge dry run`, `oidcmprec` has a user-defined compare and reconcile operation, `userdefinedcr`, that enables you to specifying conflict resolution rule arguments. Any conflict resolution rule you do not specify with `-userdefinedcr` defaults to `ignore`. The following command line uses the `userdefinedcr` operation:

```
oidcmprec operation=userdefinedcr scope=subtree \
          base="'dc=com' 'dc=org'" \
          source=myhost1.mycom.com:3060 \
          destination=myhost2.mycom.com:3060 \
          entos=add entod=ignore atros=add atrod=ignore \
          svatrdif=usesrc mvatrdif=usesrc dndif=ignore \
          threads=5 dnthreads=2 file=myreconcile
```

Conflict scenarios and conflict resolution rules are described in the oidcmprec command reference in *Reference for Oracle Identity Management*.

44.4.14 Known Limitations of the oidcmprec Tool

The `oidcmprec` tool has the following limitations:

- When the tool logs changes to LDIF records in the *`filename`*`.s2d.ldif` or *`filename`*`.d2s.ldif` file for deletion of a tree, it logs the parent record first, followed by its children. If you attempt to apply this change using the `ldapmodify` command-line tool, it fails, as the directory server does not allow deletion of non-leaf entry. To prevent `ldapmodify` from failing, edit the file to reorder the records before running `ldapmodify`.
- When the tool performs a delete operation on a entry, it deletes that entry and its children. The tool records that the entry was deleted, but does not log that its children were also deleted.
- The tool has limitations with respect to compound RDNs. These are RDNs that contain two or more *`attribute=attrvalue`* pairs, separated by a `+`, for example:

  ```
  uid=jpaul + cn=john paul + mail=john.paul@example.com,dc=oracle,dc=com
  ```

  If one of the directories you are comparing contains a compound RDN, when the tool suggests `modrdn`/`moddn` changes in the *`filename`*`.s2d.ldif` or *`filename`*`.d2s.ldif` file, the `deleteoldrdn` value might be incorrect.
- If you have provided a filter to `oidcmprec`, and you plan to use the output as input to `ldapmodify`, first edit the output to ensure that entries are in the correct order. In particular, if you are migrating a whole tree, ensure that the root of the tree is the first entry.

Part VI

Advanced Administration: Directory Plug-ins

This part contains the following chapters:

- Developing Plug-ins for the Oracle Internet Directory Server
- Configuring a Customized Password Policy Plug-In
- Configuring a Customized External Authentication Plug-in

45

Configuring a Customized Password Policy Plug-In

Oracle Internet Directory uses plug-ins to add password value checking to its other password policy management capabilities. These plug-ins enable you to verify that, for example, a new or modified password has the specified minimum length. You can customize password value checking to meet your own requirements.
The following topics describe how to install, configure, and enable a customized password policy plug-in:

- Configuring a Customized Password Policy Plug-in
- Managing a Customized Password Policy Plug-in

45.1 Configuring a Customized Password Policy Plug-in

When a user wants to add or modify a password, customized password value checking takes place as follows:

1. The client sends the directory server either an ldapadd or ldapmodify request.
2. Before the directory server makes the addition or modification, it passes the password value to the plug-in.
3. The plug-in
   a. Parses the entry
   b. Captures the `userpassword` attribute value in clear text
   c. Implements whatever password value checking you have specified
4. If the password meets the specification, then the plug-in notifies the directory server accordingly, and the directory server makes the addition or modification.

   Otherwise, the plug-in sends one of the following error messages to the directory server, which, in turn, passes it to the client.

   ```
   ldap_add: UnKnown Error Encountered
   ldap_add: additional info: PASSWORD POLICY VIOLATION:0000X, less than 8 chars

   ldap_add: UnKnown Error Encountered
   ldap_add: additional info: PASSWORD POLICY VIOLATION:0000X, contains dictionary word
   ```

   The same logic applies to the PRE ldapmodify plug-in.

The various kinds of value checks that a password policy plug-in could perform include:

- Minimum and maximum number of alphabetic characters
- Maximum number of numeric characters
- Minimum and maximum number of punctuation characters
- Maximum number of consecutive characters
- Maximum number of instances of any character

- Whether it is a dictionary word

45.2 Managing a Customized Password Policy Plug-in

The example in this section uses the PL/SQL program, `pluginpkg.sql`.

The example in this section uses the PL/SQL program, `pluginpkg.sql`. Sample PL/SQL Package pluginpkg.sql Contents. In general, this package contains:

- Two plug-in modules: `pre_add` and `pre_modify`
- One value checking function, `isGoodPwd`, which verifies that a password meets the minimum length requirement of eight characters and that it does not contain a dictionary word that is longer than four characters

Thus, in this example, if you try to add a user with the `userpassword` value less than eight characters, then the request is rejected. Similarly, if you try to modify a user password, and the new password value is less than eight characters, then the request is rejected. Also, if you try to add or modify a user with the `userpassword` supersunday, the password is rejected because super and sunday are dictionary words.

The dictionary is a list of words longer than four characters, initially stored in a file called words.txt. Before we implement the plug-in, we set up a database table and store the words into the table. To set up the table we use `create.sql`, which has the following contents:

```
drop table mydic;
create table mydic (word varchar2(1024));
commit;
exit;
```

Then we load the words into the table using the sqlldr command:

```
sqlldr control=words.txt userid=ods/ods_password
```

This section contains the following topics:

- Loading and Registering the PL/SQL Program
- Coding the Password Policy Plug-in
- Debugging the Password Policy Plug-in
- Sample PL/SQL Package pluginpkg.sql Contents

45.2.1 Loading and Registering the PL/SQL Program

This section describes the procedure to load and register the PL/SQL program.

After you implement the standalone value checking PL/SQL program, do the following:.

1. Load the plug-in package into the database. In this example, we enter:

   ```
   sqlplus ods @pluginpkg.sql
   ```

2. Register the plug-in. This example uses a file named `pluginreg.dat`, which contains the following:

   ```
   ### add plugin ###
   dn: cn=pre_add_plugin,cn=plugin,cn=subconfigsubentry
   objectclass:orclPluginConfig
   objectclass:top
   orclpluginname:pwd_plugin
   ```

```
orclplugintype:operational
orclplugintiming:pre
orclpluginldapoperation:ldapadd
orclpluginenable:1
orclpluginversion:1.0.1
cn:pre_add_plugin
orclpluginsubscriberdnlist:dc=com;o=IMC ,c=US

### modify plugin ###
dn: cn=pre_mod_plugin,cn=plugin,cn=subconfigsubentry
objectclass:orclPluginConfig
objectclass:top
orclpluginname:pwd_plugin
orclplugintype:operational
orclplugintiming:pre
orclpluginldapoperation:ldapmodify
orclpluginenable:1
orclpluginversion:1.0.1
cn:pre_mod_plugin
orclpluginsubscriberdnlist:dc=com;o=IMC ,c=US
orclpluginattributelist:userpassword
```

Note that, in this plug-in, we let the directory server know that there are two plug-in modules to invoke when it receives ldapadd or ldapmodify requests. We use `orclpluginsubscriberdnlist:dc=com;o=IMC,c=US` so that the plug-in is invoked ONLY if the target entry is under `dc=com` or `o=IMC,c=US`.

To add this file to the directory, enter the following:

```
ldapadd -p portnum -h hostname -D cn=orcladmin -q -v -f pluginreg.dat
```

45.2.2 Coding the Password Policy Plug-in

You can use standard PL/SQL character functions to process the password value.

Download any PL/SQL program that can do regular expression. The important thing is to integrate the value checking functions with your plug-in modules.

45.2.3 Debugging the Password Policy Plug-in

First you enable the directory server plug-in to help examine the process and content of plug-ins.

To setup the directory server plug-in debugging, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdsu.pls
```

To enable directory server plug-in debugging, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdon.pls
```

To disable directory server plug-in debugging, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdof.pls
```

To show directory server plug-in debugging messages, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdsh.pls
```

To delete directory server plug-in debugging messages, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdde.pls
```

45.2.4 Sample PL/SQL Package pluginpkg.sql Contents

The script `pluginpkg.sql`, as used in this example, contains the following:

```
CREATE OR REPLACE PACKAGE pwd_plugin AS

PROCEDURE pre_add (ldapplugincontext IN  ODS.plugincontext,
           dn       IN  VARCHAR2,
           entry    IN  ODS.entryobj,
           rc       OUT INTEGER,
           errormsg OUT VARCHAR2
           );

PROCEDURE pre_modify (ldapplugincontext IN  ODS.plugincontext,
              dn       IN  VARCHAR2,
              mods     IN  ODS.modlist,
              rc       OUT INTEGER,
              errormsg OUT VARCHAR2
              );

-- Function: isGoodPwd
-- Parameter: inpwd
-- Purpose: 1. check if the password is at least
--          8 characters long
--          2. check if the password contains a
--          dictionary word (longer than 4 characters)

FUNCTION isGoodPwd(inpwd IN VARCHAR2)
  RETURN INTEGER;


END pwd_plugin;
/

show error

CREATE OR REPLACE PACKAGE BODY pwd_plugin AS

FUNCTION isGoodPwd(inpwd IN VARCHAR2)
  RETURN INTEGER
  IS
     i NUMBER;
     ret NUMBER DEFAULT 1;
     minpwdlen NUMBER DEFAULT 8;
     len       NUMBER DEFAULT 0;
     lcount    NUMBER DEFAULT 0;
     matched   VARCHAR2(1024) DEFAULT NULL;

     CURSOR c1 IS
     SELECT word FROM mydic WHERE length(word) > 4
     AND instr(lower(inpwd), lower(word), 1, 1) > 0;

BEGIN
   plg_debug( '=== begin of ISGOODPWD ===');
   plg_debug( 'password = ' || inpwd);
   len := LENGTH(inpwd);
   plg_debug( 'password length = ' || len);
```

```
   IF len < minpwdlen THEN
      RETURN 0;
    ELSE
      OPEN c1;
      LOOP
        FETCH c1 INTO matched;
        EXIT WHEN c1%NOTFOUND;
        lcount := lcount + 1;
      END LOOP;
      plg_debug( 'count = ' || lcount);
      IF lcount > 0 THEN
        RETURN 2;
      ELSE
        RETURN ret;
      END IF;
   END IF;

   plg_debug( '=== end of ISGOODPWD ===');

EXCEPTION
   WHEN OTHERS THEN
      plg_debug( 'Exception in isGoodPwd(). Error code is ' || TO_CHAR(SQLCODE));
      plg_debug( '   ' || Sqlerrm);
      RETURN 0;
END;


PROCEDURE pre_add (ldapplugincontext IN  ODS.plugincontext,
           dn       IN  VARCHAR2,
           entry    IN  ODS.entryobj,
           rc       OUT INTEGER,
           errormsg OUT VARCHAR2
           )
  IS
     inpwd VARCHAR2(256) DEFAULT NULL;
     ret   NUMBER        DEFAULT 1;
BEGIN
   plg_debug( '=== begin of PRE_ADD_PLUGIN ===');
   plg_debug( 'dn = ' || dn);

   plg_debug( 'entry obj ' || ':entryname = ' || entry.entryname);

   FOR l_counter1 IN 1..entry.attr.COUNT LOOP
      plg_debug( 'attrname[' || l_counter1 || '] = ' ||
         entry.attr(l_counter1).attrname);
      FOR l_counter2 IN 1..entry.attr(l_counter1).attrval.COUNT LOOP
     plg_debug( entry.attr(l_counter1).attrname ||
            '[' || l_counter1 || ']' ||
            '.val[' || l_counter2 || '] = ' ||
            entry.attr(l_counter1).attrval(l_counter2));
      END LOOP;

      IF entry.attr(l_counter1).attrname = 'userpassword' THEN
     inpwd := entry.attr(l_counter1).attrval(1);
     -- assuming only one attr val for userpassword
      END IF;

   END LOOP;

   IF (inpwd IS NOT NULL) THEN
      ret := isGoodPwd(inpwd);
   END IF;
```

```
   IF (inpwd IS NULL OR ret = 0) THEN
      rc := 1;
      errormsg := 'PASSWORD POLICY VIOLATION:0000X, less than 8 chars';
      plg_debug( ' we got an invalid password, too short ');
    ELSIF (ret = 2) THEN
      rc := 1;
      errormsg := 'PASSWORD POLICY VIOLATION:0000X, contains dictionary word';
      plg_debug( ' we got an invalid password, dictionary word ');
    ELSE
      plg_debug( ' we got a good password ');
      rc := 0;
      errormsg := 'no pre_mod plguin error msg';
   END IF;

   plg_debug( '=== end of PRE_ADD_PLUGIN ===');

EXCEPTION
   WHEN OTHERS THEN
      plg_debug( 'Exception in PRE_ADD plugin. Error code is ' || TO_CHAR(SQLCODE));
      plg_debug( '   ' || Sqlerrm);
      rc := 1;
      errormsg := 'exception: pre_add plguin';
END;

PROCEDURE pre_modify (ldapplugincontext IN  ODS.plugincontext,
              dn       IN  VARCHAR2,
              mods     IN  ODS.modlist,
              rc       OUT INTEGER,
              errormsg OUT VARCHAR2
              )
  IS
     old_passwd VARCHAR2(256) DEFAULT NULL;
     new_passwd VARCHAR2(256) DEFAULT NULL;
     ret        NUMBER        DEFAULT 1;

BEGIN
   plg_debug( '=== begin of PRE_MOD_PLUGIN ===');
   plg_debug( dn);

   FOR l_counter1 IN 1..mods.COUNT LOOP
      IF (mods(l_counter1).operation = 2) AND
    (mods(l_counter1).type = 'userpassword') THEN

     FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
        new_passwd := mods(l_counter1).vals(l_counter2).val;
     END LOOP;
      END IF;

      IF (mods(l_counter1).operation = 0) AND
    (mods(l_counter1).type = 'userpassword') THEN

     FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
        new_passwd := mods(l_counter1).vals(l_counter2).val;
     END LOOP;
      END IF;

      IF (mods(l_counter1).operation = 1) AND
    (mods(l_counter1).type = 'userpassword') THEN

     FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
        old_passwd := mods(l_counter1).vals(l_counter2).val;
```

```
      END LOOP;
       END IF;
   END LOOP;

   plg_debug(' new password: ' || new_passwd);
   plg_debug(' old password: ' || old_passwd);

   IF (new_passwd IS NOT NULL) THEN
      ret := isGoodPwd(new_passwd);
   END IF;

   IF (new_passwd IS NULL OR ret = 0) THEN
      rc := 1;
      errormsg := 'PASSWORD POLICY VIOLATION:0000X, less than 8 chars';
      plg_debug( ' we got an invalid password, too short ');
    ELSIF (ret = 2) THEN
      rc := 1;
      errormsg := 'PASSWORD POLICY VIOLATION:0000X, contains dictionary word';
      plg_debug( ' we got an invalid password, dictionary word ');
    ELSE
      plg_debug( ' we got a good password ');
      rc := 0;
      errormsg := 'no pre_mod plguin error msg';
   END IF;

   plg_debug( '=== end of PRE_MOD_PLUGIN ===');

EXCEPTION
   WHEN OTHERS THEN
      plg_debug( 'Exception in PRE_MODIFY plugin. Error code is ' || TO_CHAR(SQLCODE));
      plg_debug( '    ' || Sqlerrm);
      rc := 1;
      errormsg := 'exception: pre_mod plguin';
END;

END pwd_plugin;
/
show error


EXIT;
```

46

Developing Plug-ins for the Oracle Internet Directory Server

The following topics describe how to create a plug-in, register a plug-in using LDAP command-line utilities, and manage plug-ins using Oracle Directory Services Manager (ODSM) and Oracle Enterprise Manager Fusion Middleware Control:

- Overview of Oracle Internet Directory Server Plug-in Framework
- Creating a Plug-in
- Registering a Plug-in From the Command Line
- Managing Plug-ins by Using Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control

46.1 Overview of Oracle Internet Directory Server Plug-in Framework

A server plug-in is a customized program that can be used to extend the capabilities of the Oracle Internet Directory server.

A server plug-in can be a PL/SQL package, Java program or package, shared object or library, or a dynamic link library on Windows. Each plug-in has a configuration entry in the Oracle Internet Directory Server. The configuration entry specifies the conditions for invoking the plug-in. The conditions for invoking a plug-in include:

- An LDAP operation, such as `ldapbind` or `ldapmodify`
- A timing, relative to the LDAP operation, such as pre_bind or post_modify

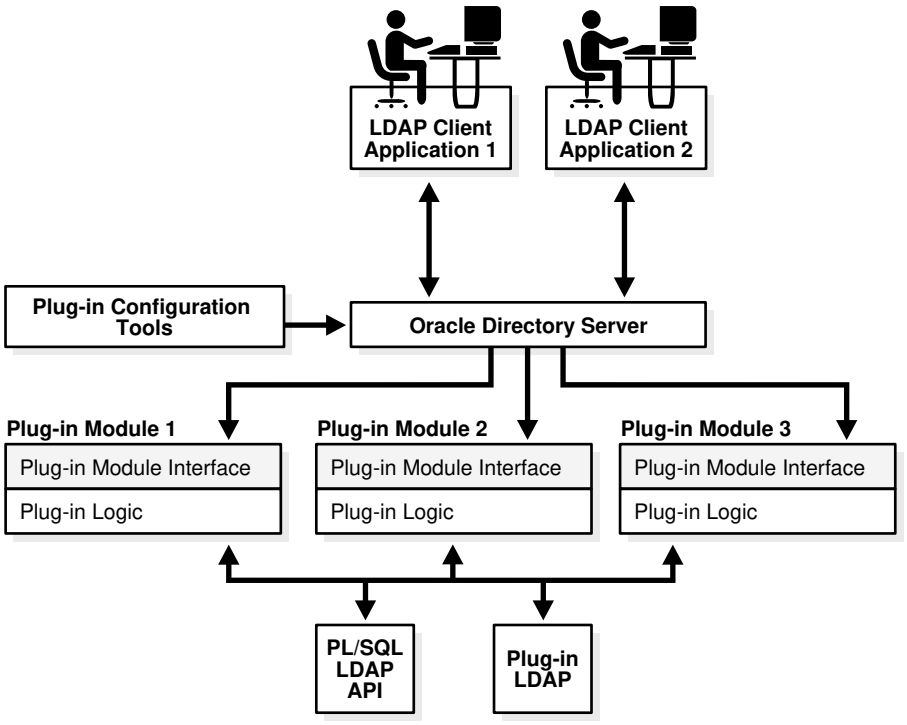
Directory server plug-ins can provide the directory server with the following kinds of added functionality, to mention just a few:

- Validate data before the directory server performs an operation on it
- Perform specified actions after the server performs an operation
- Define password policies
- Authenticate users through external credential stores

On startup, the directory server loads your plug-in configuration and library. Then, when it processes requests, it calls your plug-in functions whenever the specified event takes place.

In Figure 46-1, LDAP clients, each using a separate application, send information to and receive it from the Oracle directory server. Plug-in configuration tools likewise send information to the directory server. The directory server sends data to Plug-in Module 1, Plug-in Module 2, and Plug-in Module 3. Each plug-in module has both a plug-in module interface and plug-in logic. Each plug-in module sends information to and receives it from the LDAP API and the Plug-in LDAP.

Figure 46-1 Oracle Internet Directory Plug-in Framework



The work that plug-ins perform depends on whether they execute before, after, or in addition to normal directory server operations.

This section contains the following topics:

- Supported Languages for Server Plug-ins
- Prerequisites to Develop Server Plug-ins
- Benefits of Using Server Plug-ins
- Guidelines for Designing Server Plug-ins
- Using the Server Plug-in Framework
- LDAP Operations Supported by Oracle Internet Directory
- Understanding LDAP Timings Supported by Oracle Internet Directory
- Using Plug-ins in a Replication Environment
- Using Plug-ins in a Replication Environment
- Modifying JVM Options for Server Plug-ins

46.1.1 Supported Languages for Server Plug-ins

Oracle Internet Directory supports plug-ins in Java and in PL/SQL.

This chapter provides information common to Java and PL/SQL plug-ins. PL/SQL Server Plug-in Developer's Reference provides additional information specific to PL/SQL plug-ins and Java Server Plug-in Developer's Reference provides additional information specific to Java plug-ins.

46.1.2 Prerequisites to Develop Server Plug-ins

To develop Oracle Internet Directory plug-ins, you should be familiar with the following topics:

- Generic LDAP concepts
- Oracle Internet Directory
- Oracle Internet Directory integration with Oracle Application Server

You should have programming skills in one of the following areas:

- SQL, PL/SQL, and database RPCs
- Java

46.1.3 Benefits of Using Server Plug-ins

Some of the ways you can extend LDAP operations by using plug-ins include the following:

- You can validate data before the server performs an LDAP operation on the data.
- You can perform actions that you define after the server successfully completes an LDAP operation.
- You can define extended operations.
- You can authenticate users through external credential stores.
- You can replace an existing server module with your own server module

On startup, the directory server loads your plug-in configuration and library. It calls your plug-in functions while processing various LDAP requests.

> **See Also:**
>
> The Configuring a Customized Password Policy Plug-Inchapter about the password policy plug-in in *Administering Oracle Internet Directory*. Tise chapter contains an example of how to implement your own password value checking and place it into the Oracle Internet Directory server.

46.1.4 Guidelines for Designing Server Plug-ins

Use the following guidelines when designing plug-ins:

- Use plug-ins to guarantee that when a specific LDAP operation is performed, related actions are also performed.
- Use plug-ins only for centralized, global operations that should be invoked for the program body statement, regardless of which user or LDAP application issues the statement.
- Do not create recursive plug-ins. For example, creating a `pre_ldap_bind` plug-in that itself issues an `ldapbind` statement would cause the plug-in to execute recursively until it has run out of resources.

  Use plug-ins judiciously. They are executed every time the associated LDAP operation occurs.

46.1.5 Using the Server Plug-in Framework

The plug-in framework is the environment in which you develop, configure, and apply the plug-ins. Each individual plug-in instance is called a plug-in module.

The plug-in framework includes the following:

- Plug-in configuration tools
- Plug-in module interface
- Plug-in LDAP APIs:
  - PL/SQL package `ODS.LDAP_PLUGIN`
  - Java package `oracle.ldap.ospf`

For both languages, you follow these general steps to use the server plug-in framework:

1. Write a user-defined plug-in procedure in PL/SQL or Java.
2. Compile the plug-in module.
3. Register the plug-in module through the configuration entry interface by using either the command line or Oracle Directory Services Manager.

> **Note:**
>
> Access to external network services in database is restricted. If DBMS_LDAP package is used in PL/SQL plug-ins, make sure that proper user access controls are configured in the database to enable external network calls by the plug-ins. See Managing Fine-Grained Access in PL/SQL Packages and Types to know about PL/SQL packages and Oracle Application Security access control lists (ACL).
> Without following the procedure to configure user access controls, you may encounter "ORA-24247: network access denied by access control list (ACL)" while using OID PL/SQL plug-in.

46.1.6 LDAP Operations Supported by Oracle Internet Directory

The Oracle Internet Directory server supports plug-ins for the following LDAP operations:

- `ldapadd`
- `ldapbind`
- `ldapcompare`
- `ldapdelete`
- `ldapmoddn` (Java only)
- `ldapmodify`
- `ldapsearch`

46.1.7 Understanding LDAP Timings Supported by Oracle Internet Directory

Oracle Internet Directory supports four operation timings for plug-ins:

- pre
- post
- when
- when_replace

This section contains the following topics:

- About Pre-Operation Server Plug-ins
- About Post-Operation Server Plug-ins
- About When-Operation Server Plug-ins
- About When_Replace-Operation Server Plug-ins

46.1.7.1 About Pre-Operation Server Plug-ins

The server calls pre-operation plug-in modules before performing the LDAP operation. The main purpose of this type of plug-in is to validate data before the data is used in the LDAP operation.

When an exception occurs in the pre-operation plug-in, one of the following occurs:

- When the return error code indicates warning status, the associated LDAP request proceeds.
- When the return code indicates failure status, the request does not proceed.

If the associated LDAP request fails later on, the directory does not roll back the committed code in the plug-in modules.

46.1.7.2 About Post-Operation Server Plug-ins

The Oracle Internet Directory server calls post-operation plug-in modules after performing an LDAP operation. The main purpose of this type of plug-in is to invoke a function after a particular LDAP operation is executed. For example, logging and notification are post-operation plug-in functions.

When an exception occurs in the post-operation plug-in, the associated LDAP operation is not rolled back.

If the associated LDAP request fails, the post plug-in is still executed.

46.1.7.3 About When-Operation Server Plug-ins

The directory calls when-operation plug-in modules while performing standard LDAP operations. A when-operation plug-in executes immediately before the server's own code for the operation. The main purpose of this type of plug-in is to augment existing operations within the same LDAP transaction. If the when-operation plug-in fails, the standard LDAP operation does not execute. If the when-operation plug-in completes successfully, but the standard LDAP operation fails, then the changes made in the plug-in are not rolled back.

You can, for example, use a when-operation plug-in with the ldapcompare operation. The directory executes its server compare code and executes the plug-in module defined by the plug-in developer.

PL/SQL when-operation plug-ins are supported in `ldapadd`, `ldapdelete`, and `ldapmodify`. Java when_operation plug-ins are supported in `ldapadd`, `ldapdelete`, `ldapmoddn`, `ldapmodify`, and `ldapsearch`.

46.1.7.4 About When_Replace-Operation Server Plug-ins

A when_replace-operation plug-in executes instead of the server's code for the operation. You can, for example, use a when_replace plug-in with the ldapcompare operation. The directory does not execute its compare code. Instead it relies on the plug-in module to perform the comparison.

PL/SQL when_replace-operation plug-ins are supported only in `ldapadd`, `ldapcompare`, `ldapdelete`, `ldapmodify`, and `ldapbind`.

Java when_replace-operation plug-ins are supported in ldapadd, ldapbind, ldapcompare, `ldapdelete`, `ldapmoddn`, `ldapmodify` and `ldapsearch`.

46.1.8 Using Plug-ins in a Replication Environment

Use caution when deploying plug-ins in a replication environment.

The following bad practices can result in an inconsistent state:

- Plug-in metadata replicated to other nodes
- Plug-in installation on some, but not all, of the participating nodes
- Implementation in the plug-in of extra checking that depends on the directory data
- Changes to directory entries by plug-in programs or other LDAP operations

You can use plug-ins that change directory entries in a replication environment if you deploy the plug-in on all nodes and configure the plug-in so that changes caused by replication do not invoke the plug-in. You do that as follows:

- Add the replica IDs of all the nodes to a group.
- Include the group as a value in the plug-in attribute `orclpluginrequestneggroup`. For example:

  ```
  orclpluginrequestneggroup: cn=pluginNegate,cn=groups,cn=oraclecontext.
  ```

  Whenever a plug-in detects that a change request has come from a member of the group `cn=pluginNegate`, the plug-in is not invoked.

46.1.9 Modifying JVM Options for Server Plug-ins

The Java server plug-ins run in a Java Virtual Machine (JVM) within the `oidldapd` server itself.

The JVM is implemented using the Java Native Interface (JNI). The `oidldapd` server passes options to the JVM when it invokes it. These JVM options are contained in the `orcljvmoptions` attribute of the DSA configuration entry. By default, the value of this attribute is `-Xmx64M`, which specifies a heap size of 64 MB. You can modify the options by changing the value of `orcljvmoptions`. Normally, you only need to do this if you add a plug-in that requires a greater heap size than the default of 64 M.

You can modify this attribute in the same way as other attributes of the DSA configuration entry, or by using Oracle Enterprise Manager Fusion Middleware Control. For more information, see Managing System Configuration Attributes by Using LDAP Tools and Managing JVM Options by Using Oracle Enterprise Manager Fusion Middleware Control.

46.2 Creating a Plug-in

This section describes the procedure to create a server plug-in.

To use the server plug-in framework, follow these steps:

1. Write a user-defined plug-in procedure in PL/SQL or Java.
2. Compile the plug-in module.
3. Register the plug-in module through the configuration entry interface by using either the command line or Oracle Directory Services Manager.

46.3 Registering a Plug-in From the Command Line

To enable the directory server to call a plug-in at the right time, you must register the plug-in with the directory server.

To register a plug-in, creatie an entry for the plug-in in the directory schema under `cn=plugin,cn=subconfigsubentry`.

This section contains the following topics:

- Object Classes and Attributes to Create a Plug-in Configuration Entry
- Adding a Plug-in Configuration Entry by Using Command-Line Tools

46.3.1 Object Classes and Attributes to Create a Plug-in Configuration Entry

This section lists and describes the object classes and attributes you can specify in a plug-in configuration.

Table 46-1 lists and describes the object classes and attributes you can specify in a plug-in configuration.

Table 46-1 Plug-in Configuration Objects and Attributes

Name	Value	Mandatory?
`objectclass`	`orclPluginConfig`	Yes
`objectclass`	`top`	No
`dn`	Plug-in entry DN	Yes
`cn`	Plug-in entry name	Yes
`orclPluginAttributeList`	A semicolon-separated list of attribute names that controls whether the plug-in takes effect. If the target attribute is included in the list, then the plug-in is invoked. Only for `ldapcompare` and `ldapmodify` plug-ins.	No
`orclPluginEnable`	`0` = disable (default) `1` = enable	No

Table 46-1 (Cont.) Plug-in Configuration Objects and Attributes

Name	Value	Mandatory?
orclPluginEntryProperties	An ldapsearch filter type value. For example, if we specify orclPluginEntryProperties: (&(objectclass=inetorgperson) (sn=Cezanne)), the plug-in is not invoked if the target entry has objectclass equal to inetorgperson and sn equal to Cezanne.	No
orclPluginIsReplace	0 = disable (default) 1 = enable For when_replace timing, enable this and set orclPluginTiming to when.	No
orclPluginKind	PL/SQL or Java (Default is PL/SQL)	No
orclPluginLDAPOperation	One of the following values: ldapcompare ldapmodify ldapbind ldapadd ldapdelete ldapsearch ldapmoddn (Java Only)	Yes
orclPluginName	Plug-in name	Yes
orclPluginFlexfield	Custom text information (Java only). To indicate a subtype, specify orclPluginFlexfield;subtypename, for example, orclPluginFlexfield;minPwdLength: 8	No
orclPluginBinaryFlexfield	Custom binary information (Java only).	No
orclPluginSecuredFlexfield	Custom text information that must never be displayed in clear text (Java only). To indicate a subtype, specify orclPluginSecuredFlexfield;subtypename, for example orclPluginSecuredFlexfield;telephonenumber1: 650.123.456. The value is stored and displayed in encrypted form. In a search result, it might appear as something like this: orclPluginSecuredFlexfield;telephonenumber1: 1291zjs8134. Be sure that Oracle Internet Directory has privacy mode enabled to ensure that users cannot retrieve this attribute in clear text. See "Privacy of Retrieved Sensitive Attributes" in *Administering Oracle Internet Directory*.	No

Table 46-1 (Cont.) Plug-in Configuration Objects and Attributes

Name	Value	Mandatory?
orclPluginRequestGroup	A semicolon-separated group list that controls if the plug-in takes effect. You can use this group to specify who can actually invoke the plug-in. For example, if you specify orclpluginrequestgroup:cn=security,cn=groups,dc=oracle,dc=com when you register the plug-in, the plug-in is not invoked unless the ldap request comes from the person who belongs to the group cn=security,cn=groups,dc=oracle,dc=com.	No
orclPluginRequestNegGroup	A semicolon-separated group list that controls if the plug-in takes effect. You can use this group to specify who cannot invoke the plug-in. For example, if you specify orclpluginrequestgroup: cn=security,cn=groups,dc=oracle,dc=com, when you register the plug-in, the plug-in is not invoked if the LDAP request comes from the person who belongs to the group cn=security,cn=groups,dc=oracle, dc=com.	No
orclPluginResultCode	An integer value to specify the ldap result code. If this value is specified, then plug-in is invoked only if the LDAP operation is in that result code scenario. This is only for the post plug-in type.	No
orclPluginShareLibLocation	File location of the dynamic linking library. If this value is not present, then Oracle Internet Directory server assumes the plug-in language is PL/SQL.	No
orclPluginSubscriberDNList	A semicolon separated DN list that controls if the plug-in takes effect. If the target DN of an LDAP operation is included in the list, then the plug-in is invoked.	No
orclPluginTiming	One of the following values: pre when post For when_replace timing, specify when and enable orclPluginIsReplace.	No
orclPluginType	One of the following values: configuration attribute password_policy syntax matchingrule **See Also:** "LDAP Operations Supported by Oracle Internet Directory".	Yes
orclPluginVersion	Supported plug-in version number	No

Table 46-1 (Cont.) Plug-in Configuration Objects and Attributes

Name	Value	Mandatory?
orclPluginClassReloadEnabled	If this value is 1, the server reloads the plug-in class every time it invokes the plug-in. If the value is 0, the server loads the class only the first time it invokes the plug-in.	

46.3.2 Adding a Plug-in Configuration Entry by Using Command-Line Tools

To add a plug-in configuration entry from the command line, create an LDIF file containing the plug-in configuration. Specify a DN under `cn=plugin,cn=subconfigsubentry`.

The following two-part LDIF file, `my_ldif_file.ldif`, creates an entry for an operation-based plug-in called `my_plugin1`:

```
dn: cn=when_comp,cn=plugin,cn=subconfigsubentry
objectclass: orclPluginConfig
objectclass: top
orclPluginName: my_plugin1
orclPluginType: configuration
orclPluginTiming: when
orclPluginLDAPOperation: ldapcompare
orclPluginEnable: 1
orclPluginVersion: 1.0.1
orclPluginIsReplace: 1
cn: when_comp
orclPluginKind: PLSQL
orclPluginSubscriberDNList: dc=COM,c=us;dc=us,dc=oracle,dc=com;dc=org,dc=us;
 o=IMC,c=US
orclPluginAttributeList: userpassword

dn: cn=post_mod_plugin, cn=plugin,cn=subconfigsubentry
objectclass: orclPluginConfig
objectclass: top
orclPluginName: my_plugin1
orclPluginType: configuration
orclPluginTiming: post
orclPluginLDAPOperation: ldapmodify
orclPluginEnable: 1
orclPluginVersion: 1.0.1
cn: post_mod_plugin
orclPluginKind: PLSQL
```

Add this file to the directory with a command similar to this:

```
ldapadd -p 3060 -h myhost -D binddn -q -f my_ldif_file.ldif
```

> **Note:**
>
> The plug-in configuration entry is not replicated. Replicating it would create an inconsistent state.

46.4 Managing Plug-ins by Using Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control

Use Oracle Directory Services Manager to register, edit, and delete plug-ins.

This section contains the following topics:

- Creating a Plug-in by Using Oracle Directory Services Manager
- Registering a Plug-in by Using Oracle Directory Services Manager
- Editing a Plug-in by Using Oracle Directory Services Manager
- Deleting a Plug-in by Using Oracle Directory Services Manager
- Managing JVM Options by Using Oracle Enterprise Manager Fusion Middleware Control

46.4.1 Creating a Plug-in by Using Oracle Directory Services Manager

This section describes the procedure to create a plug-in using Oracle Directory Services Manager.

To create a new plug-in, follow these steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Advanced**.
3. Expand **Plug-in**. Entries appear in the left panel.
4. To enable a plug-in management configuration entry, select it.
5. Click the **Create** icon.The New Plug-in window appears in the right pane.
6. Select **Plug-in Enable** if you want to enable the plug-in now.
7. Enter the **Plug-in Name**.
8. Select values for the other mandatory properties.
9. When you have finished entering the values, select **OK**. The plug-in you just created is now listed on the left side of the page.

46.4.2 Registering a Plug-in by Using Oracle Directory Services Manager

This section describes the procedure to register a plug-in using Oracle Directory Services Manager.

To register a plug-in, follow these steps:

1. Invoke Oracle Directory Services Manager and connect to the Oracle Internet Directory server as described in Invoking Oracle Directory Services Manager.
2. From the task selection bar, select **Advanced**.
3. Expand **Plug-in**. Entries appear in the left panel.
4. To enable a plug-in management configuration entry, select it. The Plug-in Management tab appears in the right pane.

5. Select **Plug-in Enable**.
6. Click **Apply**.
7. When you have finished entering the values, select **OK**. The plug-in you just created is now listed on the left side of the page.

46.4.3 Editing a Plug-in by Using Oracle Directory Services Manager

This section describes the procedure to edit a plug-in using Oracle Directory Services Manager.

To edit a plug-in entry, follow these steps:

1. From the task selection bar, select **Advanced**.
2. Expand **Plug-in**. Entries appear in the left panel.
3. To modify a plug-in management configuration entry, select it.
4. Enter changes to Mandatory Properties and Optional Properties on the right side of the page.
5. Click **Apply**.

46.4.4 Deleting a Plug-in by Using Oracle Directory Services Manager

This section describes the procedure to delete a plug-in using Oracle Internet Directory Services Manager.

To delete a plug-in, follow these steps:

1. From the task selection bar, select **Advanced**.
2. Expand **Plug-in**. Entries appear in the left panel.
3. Select the plug-in entry you want to delete.
4. Click the **Delete** icon. The plug-in entry you deleted no longer appears in the list.

46.4.5 Managing JVM Options by Using Oracle Enterprise Manager Fusion Middleware Control

JVM options are contained in the `orcljvmoptions` attribute of the DSA configuration entry. The default value is `-Xmx64M`, which sets the heap size to 64M. If you need to allocate more heap, update this attribute.

To modify this attribute by using Oracle Enterprise Manager Fusion Middleware Control, see Configuring Shared Properties. To modify it from the command line, see Managing Entries by Using LDAP Command-Line Tools.

47

Configuring a Customized External Authentication Plug-in

You can store user security credentials in a repository other than Oracle Internet Directory—for example, a database or another LDAP directory—and use these credentials for user authentication to Oracle components. You do not need to store the credentials in Oracle Internet Directory and then worry about keeping them synchronized. Authenticating a user by way of credentials stored in an external repository is called external authentication.
The following topics describe how to use a customized external authentication plug-in, how to install, configure, enable, and debug the external plug-in, and how to create a PL/SQL package:

- Overview of Customized External Authentication Plug-in
- Installing, Configuring, and Enabling the External Authentication Plug-in
- Debugging the External Authentication Plug-in
- Creating the PL/SQL Package oidexaup.sql

> **Note:**
>
> All references to Oracle Single Sign-On in this chapter refer to Oracle Single Sign-On 10g (10.1.4.3.0) or later.

47.1 Overview of Customized External Authentication Plug-in

Authentication that relies on security credentials stored in Oracle Internet Directory is called native authentication.

When a user enters her security credentials, the directory server compares them with the credentials stored in Oracle Internet Directory. If the credentials match, then the directory server authenticates the user.

By contrast, authentication that relies on security credentials stored in a directory other than Oracle Internet Directory is called external authentication. When a user enters her security credentials, the directory server compares them with the credentials stored in the other directory. This is done by using:

- A PL/SQL program that does the external authentication work
- An external authentication plug-in that invokes this PL/SQL program

47.2 Installing, Configuring, and Enabling the External Authentication Plug-in

The package, `oidexaup.sql`, is used for installing the external authentication plug-in PL/SQL package.

This example uses the PL/SQL program, `oidexaup.sql`. Creating the PL/SQL Package oidexaup.sql describes this program. It contains:

- Two plug-ins: namely, `when_compare_replace` and `when_modify_replace`
- One utility function: namely, `get_nickname`

The integrated package is the plug-in package, `OIDEXTAUTH`. It can also serve as a template to modify according to the requirements of your deployment.

To install, configure, and enable the external authentication plug-in, follow these steps:

1. Implement your standalone external authentication PL/SQL program. For example, if you want to authenticate users by using user names and passwords, then you should have a PL/SQL program which takes these two parameters.

   In our sample code, `oidexaup.sql`, `auth_external` is the program package name, and `authenticate_user` is the function that does the authentication. you must make sure that this standalone program is working properly before you move on to next steps.

2. Integrate this standalone program into the plug-in modules.
3. Load the plug-in package into database. In this example, we enter:

   ```
   sqlplus ods/odspwd @oidexaup.sql
   ```

4. Register the plug-ins. Do this by creating and uploading an LDIF file that provides the directory server with the necessary information to invoke the plug-in.
5. This example uses a file named oidexauth.ldif, which contains the following:

   ```
   dn: cn=whencompare,cn=plugin,cn=subconfigsubentry
   objectclass:orclPluginConfig
   objectclass:top
   orclpluginname:oidextauth
   orclplugintype:configuration
   orclplugintiming:when
   orclpluginldapoperation:ldapcompare
   orclpluginenable:1
   orclpluginversion:1.0.1
   orclPluginIsReplace:1
   cn:whencompare
   orclpluginsubscriberdnlist:dc=com;o=IMC,c=US
   orclpluginattributelist:userpassword
   orclpluginrequestgroup:$prgdn

   dn: cn=whenmodify,cn=plugin,cn=subconfigsubentry
   objectclass:orclPluginConfig
   objectclass:top
   orclpluginname:oidextauth
   orclplugintype:configuration
   orclplugintiming:when
   orclpluginldapoperation:ldapmodify
   orclpluginenable:1
   orclpluginversion:1.0.1
   ```

```
orclPluginIsReplace:1
cn:whenmodify
orclpluginsubscriberdnlist:dc=com;o=IMC,c=US
orclpluginattributelist:userpassword
orclpluginrequestgroup:$prgdn
```

In this file, we notify the directory server that, whenever there is an ldapcompare or ldapmodify request, there are two plug-ins to be invoked.

We use `orclpluginsubscriberdnlist:dc=com;o=IMC,c=US` so that plug-ins are ONLY invoked if the target entry is under `dc=com` or `o=IMC,c=US`.

Replace `$prgdn` with the plug-in request group DN. This is an optional, recommended security feature. For integrating with Oracle Single Sign-On, this value is a required field. Only members of the group entered can invoke the plug-ins. You may enter multiple groups. Use a semicolon to separate entries.

The recommended defaults are:
`cn=OracleUserSecurityAdmins,cn=Groups,cn=OracleContext` and `cn=OracleDASAdminGroup,cn=Groups,cn=OracleContext,o=default_subscriber,dc=com`. Note that the Oracle Single Sign-On server is a member of the first group. Also, be sure to replace `o=default_subscriber` with the correct value for your deployment environment.

To add this file to the directory, enter the following:

```
ldapadd -p portnum -h hostname -D cn=orcladmin -q -v \
        -f oidexauth.ldif
```

Now, everything should be ready. Use the ldapcompare command-line tool to verify that the plug-in and authentication program are working properly before you try to authenticate the user from Oracle Single Sign-On.

In our example, we also provide the plug-in code for externally modifying user password.

47.3 Debugging the External Authentication Plug-in

You must enable directory server plug-in to help you to examine the process and content of plug-ins.

To setup directory server plug-in debugging, execute the following command:

```
sqlplus ods  @$ORACLE_HOME/ldap/admin/oidspdsu.sql
```

To enable directory server plug-in debugging, execute the following command:

```
sqlplus ods  @$ORACLE_HOME/ldap/admin/oidspdon.sql
```

To disable directory server plug-in debugging, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdof.sql
```

To show directory server plug-in debugging messages, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdsh.sql
```

To delete directory server plug-in debugging messages, execute the following command:

```
sqlplus ods @$ORACLE_HOME/ldap/admin/oidspdde.sql
```

47.4 Creating the PL/SQL Package oidexaup.sql

You use the script `oidexaup.sql`, as used in this example, to create the PL/SQL package.

The PL/SQL package contains the following:

```
CREATE OR REPLACE PACKAGE OIDEXTAUTH AS

  PROCEDURE when_compare_replace (ldapplugincontext IN  ODS.plugincontext,
                                  result             OUT INTEGER,
                                  dn                 IN  VARCHAR2,
                                  attrname           IN  VARCHAR2,
                                  attrval            IN  VARCHAR2,
                                  rc                 OUT INTEGER,
                                  errormsg           OUT VARCHAR2
                                  );

  PROCEDURE when_modify_replace (ldapplugincontext IN  ODS.plugincontext,
                                 dn                IN  VARCHAR2,
                                 mods              IN  ODS.modlist,
                                 rc                OUT INTEGER,
                                 errormsg          OUT VARCHAR2
                                 );

  FUNCTION get_nickname (dn         IN  VARCHAR2,
                         my_session IN  DBMS_LDAP.session)
    RETURN VARCHAR2;

END OIDEXTAUTH;
/

SHOW ERROR

CREATE OR REPLACE PACKAGE BODY OIDEXTAUTH AS

  -- We use this function to convert the dn to nickname.
  -- When OID server receives the ldapcompare request, it
  -- only has the dn information. We need to use DBMS_LDAP_UTL
  -- package to find out the nickname attribute value of
  -- the entry.

  FUNCTION get_nickname (dn         IN VARCHAR2,
                         my_session IN DBMS_LDAP.session)
    RETURN VARCHAR2
    IS
       my_pset_coll        DBMS_LDAP_UTL.PROPERTY_SET_COLLECTION;
       my_property_names   DBMS_LDAP.STRING_COLLECTION;
       my_property_values  DBMS_LDAP.STRING_COLLECTION;

       user_handle         DBMS_LDAP_UTL.HANDLE;
       user_id             VARCHAR2(2000);
       user_type           PLS_INTEGER;
       user_nickname       VARCHAR2(256) DEFAULT NULL;

       my_attrs            DBMS_LDAP.STRING_COLLECTION;
       retval              PLS_INTEGER;

  BEGIN
     plg_debug( '=== Beginning of get_nickname() === ');
     user_type     := DBMS_LDAP_UTL.TYPE_DN;
```

```
   user_id       := dn;

   retval := DBMS_LDAP_UTL.create_user_handle(user_handle, user_type, user_id);

   plg_debug('create_user_handle() Returns ' || To_char(retval));

   retval := DBMS_LDAP_UTL.get_user_properties(my_session,
                                               user_handle,
                                               my_attrs,
                                               DBMS_LDAP_UTL.NICKNAME_PROPERTY,
                                               my_pset_coll);

   plg_debug( 'get_user_properties() Returns ' || To_char(retval));

   IF my_pset_coll.COUNT > 0 THEN
      FOR i IN my_pset_coll.first .. my_pset_coll.last LOOP
         retval := DBMS_LDAP_UTL.get_property_names(my_pset_coll(i),
                                                    my_property_names);
         IF my_property_names.COUNT > 0 THEN
            FOR j IN my_property_names.first .. my_property_names.last LOOP
               retval := DBMS_LDAP_UTL.get_property_values(my_pset_coll(i),
                                                           my_property_names(j),
                                                           my_property_values);
               IF my_property_values.COUNT > 0 THEN
                  FOR k IN my_property_values.FIRST..my_property_values.LAST LOOP
                     user_nickname := my_property_values(k);
                     plg_debug( 'user nickname = ' || user_nickname);
                  END LOOP;
               END IF;
            END LOOP;
         END IF; -- IF my_property_names.count > 0
      END LOOP;
   END IF; -- If my_pset_coll.count > 0

   plg_debug( 'got user_nickname: ' || user_nickname);

   -- Free my_properties
   IF my_pset_coll.count > 0 then
      DBMS_LDAP_UTL.free_propertyset_collection(my_pset_coll);
   END IF;

   DBMS_LDAP_UTL.free_handle(user_handle);

   RETURN user_nickname;

EXCEPTION
   WHEN OTHERS THEN
      plg_debug('Exception in get_nickname. Error code is ' || to_char(sqlcode));
      plg_debug('   ' || Sqlerrm);
      RETURN NULL;
END;


PROCEDURE when_compare_replace (ldapplugincontext IN  ODS.plugincontext,
                                result            OUT INTEGER,
                                dn                IN  VARCHAR2,
                                attrname          IN  VARCHAR2,
                                attrval           IN  VARCHAR2,
                                rc                OUT INTEGER,
                                errormsg          OUT VARCHAR2
                                )
  IS
```

```
    retval pls_integer;
    lresult BOOLEAN;

    my_session          DBMS_LDAP.session;
    my_property_names   DBMS_LDAP.STRING_COLLECTION;
    my_property_values  DBMS_LDAP.STRING_COLLECTION;
    my_attrs            DBMS_LDAP.STRING_COLLECTION;
    my_pset_coll        DBMS_LDAP_UTL.PROPERTY_SET_COLLECTION;
    user_handle         DBMS_LDAP_UTL.HANDLE;

    user_id             VARCHAR2(2000);
    user_type           PLS_INTEGER;
    user_nickname       VARCHAR2(60);
    remote_dn           VARCHAR2(256);

    i                   PLS_INTEGER;
    j                   PLS_INTEGER;
    k                   PLS_INTEGER;

BEGIN
   plg_debug( '=== Begin of WHEN-COMPARE-REPLACE plug-in');
   plg_debug( 'DN = ' || dn);
   plg_debug( 'Attr = ' || attrname);
   --plg_debug( 'Attrval = ' || attrval);

   DBMS_LDAP.USE_EXCEPTION := FALSE;
   errormsg := 'No error msg';
   rc := 0;

   -- converting dn to nickname
   my_session := LDAP_PLUGIN.init(ldapplugincontext);
   plg_debug( 'ldap_session =' || RAWTOHEX(SUBSTR(my_session,1,8)));

   retval := LDAP_PLUGIN.simple_bind_s(ldapplugincontext, my_session);
   plg_debug( 'simple_bind_res =' || TO_CHAR(retval));

   user_nickname := get_nickname(dn, my_session);
   plg_debug( 'user_nickname =' || user_nickname);

   -- unbind from the directory
   retval := DBMS_LDAP.unbind_s(my_session);
   plg_debug( 'unbind_res Returns ' || To_char(retval));

   IF (user_nickname IS NULL) THEN
      result := 32;
      errormsg := 'Can''t find the nickname';
      plg_debug( 'Can''t find the nickname');
      RETURN;
   END IF;

   plg_debug( '=== Now go to extauth ');

   BEGIN
      retval := auth_external.authenticate_user(user_nickname, attrval);
      plg_debug( 'auth_external.authenticate_user() returns = ' || 'True');
      result := 6; -- compare result is TRUE
   EXCEPTION
      WHEN OTHERS THEN
         result := 5; -- compare result is FALSE
         plg_debug( 'auth_external.authenticate_user() returns = ' || 'False');
         RETURN;
   END;
```

```
   plg_debug( '=== End of WHEN-COMPARE-REPLACE plug-in');
EXCEPTION
   WHEN OTHERS THEN
      rc := 1;
      errormsg := 'Exception: when_compare_replace plugin';
      plg_debug( 'EXCEPTION: ' || retval);
      plg_debug('Exception in when_compare. Error code is ' || to_char(sqlcode));
      plg_debug('   ' || Sqlerrm);
END;


PROCEDURE when_modify_replace (ldapplugincontext IN  ODS.plugincontext,
                               dn                IN  VARCHAR2,
                               mods              IN  ODS.modlist,
                               rc                OUT INTEGER,
                               errormsg          OUT VARCHAR2
                               )
  IS
     retval pls_integer;
     lresult BOOLEAN;

     my_session          DBMS_LDAP.SESSION;
     my_property_names   DBMS_LDAP.STRING_COLLECTION;
     my_property_values  DBMS_LDAP.STRING_COLLECTION;
     my_attrs            DBMS_LDAP.STRING_COLLECTION;
     my_modval           DBMS_LDAP.BERVAL_COLLECTION;
     my_pset_coll        DBMS_LDAP_UTL.PROPERTY_SET_COLLECTION;
     user_handle         DBMS_LDAP_UTL.HANDLE;

     l_mod_array         RAW(32);
     user_id             VARCHAR2(2000);
     user_type           PLS_INTEGER;
     user_nickname       VARCHAR2(2000);
     old_passwd          VARCHAR2(60) DEFAULT NULL;
     new_passwd          VARCHAR2(60) DEFAULT NULL;
     remote_dn           VARCHAR2(256);

     i                   PLS_INTEGER;
     j                   PLS_INTEGER;
     k                   PLS_INTEGER;

BEGIN
   plg_debug( '=== Begin of WHEN-MODIFY-REPLACE plug-in');
   DBMS_LDAP.USE_EXCEPTION := FALSE;
   user_type     := DBMS_LDAP_UTL.TYPE_DN;
   user_id       := dn;

   -- converting dn to nickname
   my_session := LDAP_PLUGIN.init(ldapplugincontext);
   plg_debug( 'ldap_session =' || RAWTOHEX(SUBSTR(my_session,1,8)));

   retval := LDAP_PLUGIN.simple_bind_s(ldapplugincontext, my_session);
   plg_debug( 'simple_bind_res =' || TO_CHAR(retval));

   user_nickname := get_nickname(dn, my_session);
   plg_debug( 'user_nickname =' || user_nickname);

   -- unbind from the directory
   retval := DBMS_LDAP.unbind_s(my_session);

   FOR l_counter1 IN 1..mods.COUNT LOOP
```

```
      IF (mods(l_counter1).operation = 2) AND
        (mods(l_counter1).type = 'userpassword') THEN

         FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
            new_passwd := mods(l_counter1).vals(l_counter2).val;
         END LOOP;
      END IF;

      IF (mods(l_counter1).operation = 0) AND
        (mods(l_counter1).type = 'userpassword') THEN

         FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
            new_passwd := mods(l_counter1).vals(l_counter2).val;
         END LOOP;
      END IF;

      IF (mods(l_counter1).operation = 1) AND
        (mods(l_counter1).type = 'userpassword') THEN

         FOR l_counter2 IN 1..mods(l_counter1).vals.COUNT LOOP
            old_passwd := mods(l_counter1).vals(l_counter2).val;
         END LOOP;
      END IF;
   END LOOP;

   IF new_passwd IS NOT NULL AND old_passwd IS NOT NULL THEN
     BEGIN
        auth_external.change_passwd(user_nickname, old_passwd, new_passwd);
     EXCEPTION
        WHEN OTHERS THEN
           rc := 1;
           plg_debug( 'auth_external.change_passwd() raised exception.');
           errormsg := 'auth_external.change_passwd() raised exception.';
           RETURN;
     END;
    ELSIF new_passwd IS NOT NULL AND old_passwd IS NULL THEN
     BEGIN
        auth_external.reset_passwd(user_nickname, new_passwd);
     EXCEPTION
        WHEN OTHERS THEN
           plg_debug( 'auth_external.reset_passwd() raised exception.');
           rc := 1;
           errormsg := 'auth_external.reset_passwd() raised exception.';
           RETURN;
     END;
    ELSE
           rc := 1;
           errormsg := 'PLG_Exception. Not enough info to change passwd.';
   END IF;

   plg_debug( 'external change password succeed');
   rc := 0;
   errormsg := 'No when_mod_replace plguin error msg';

   retval := DBMS_LDAP.unbind_s(my_session);

   plg_debug( 'End of WHEN-MODIFY-REPLACE');
   --COMMIT;
EXCEPTION
   WHEN others THEN
      rc := 1;
      errormsg := 'PLG_Exception: when_modify_replace plguin';
```

```
        plg_debug('Exception in when_modify. Error code is ' || to_char(sqlcode));
        plg_debug('    ' || Sqlerrm);
  END;

END OIDEXTAUTH;
/
SHOW ERRORS
--list


EXIT;
```

A
Appendixes

Appendixes cover the difference between the current and the previous release, the various commands used and their syntaxes, troubleshooting Oracle Internet Directory and other references used for managing and performing important functions in Oracle Internet Directory.

This section contains the following topics:

- Differences Between 11*g* and 12*c*
- Managing Oracle Internet Directory Instances by Using OIDCTL
- How Replication Works
- Java Server Plug-in Developer's Reference
- PL/SQL Server Plug-in Developer's Reference
- The LDAP Filter Definition
- The Access Control Directive Format
- Globalization Support in the Directory
- Setting up Access Controls for Creation and Search Bases for Users and Groups
- Searching the Directory for User Certificates
- Adding a Directory Node by Using the Database Copy Procedure
- RFCs Supported by Oracle Internet Directory
- Managing Oracle Directory Services Manager's Java Key Store
- Starting and Stopping the Oracle Stack
- Performing a Rolling Upgrade
- Troubleshooting Oracle Internet Directory

A.1 Differences Between 11*g* and 12*c*

This following topics lists the major differences between Oracle Internet Directory Release 11g and 12c Release 2(12.2.1.3.0):

- Overview of Instance Creation and Process Management
- About Assigning SSL and non-SSL Ports
- Changed Path Names in 12c Configuration and Log Files
- About Configuring Audit Framework Using Oracle Enterprise Manager Fusion Middleware Control
- Updated Server Chaining
- About Setting Up and Managing LDAP-Based Replication
- About Java Containers

A.1.1 Overview of Instance Creation and Process Management

Understand about the differences in the instance creation between 11g and 12c Oracle Internet Directory.

This section contains the following topics:

- Creating 11g Oracle Internet Directory Instance
- Creating 12c Oracle Internet Directory Instance
- Starting and Stopping 11g Replication Server
- About Monitoring and Reporting the Status of Oracle Internet Directory Processes to OPMN by 11g OIDMON

> **See Also:**
>
> - Understanding Process Control of Oracle Internet Directory Components
> - Managing Oracle Internet Directory Instances.

A.1.1.1 Creating 11g Oracle Internet Directory Instance

In 11g Release 1, the procedure for creating an instance has changed. Configuration information for an Oracle Internet Directory instance resides in an instance-specific configuration entry.

The instance-specific configuration entry, which has a DN of the form:

```
cn=componentname,cn=osdldapd,cn=subconfigsubentry
```

where `componentname` is the name of a Oracle Fusion Middleware system component of `Type=OID`, for example, `oid1`. You do not manually create an instance-specific configuration entry. Instead, you create a Oracle Fusion Middleware component of `Type=OID`. Creating the Oracle Internet Directory component automatically generates an instance-specific configuration entry.

> **Note:**
>
> The entry in `configset0` still exists in 11g, but it is read-only and used to store default attribute values for seeding new instance-specific configuration entries.

The first Oracle Internet Directory system component is created during installation. The first Oracle Internet Directory system component, `oid1` by default, is created during installation with the Oracle instance name `asinst_1` by default. The corresponding configuration entry for this component is `cn=oid1,cn=osdldapd,cn=subconfigsubentry`. There are two ways to create an additional Oracle Internet Directory instance:

- Adding another component of `Type=OID` by using `opmnctl createcomponent`. For example:

```
opmnctl createcomponent -componentType OID \
   -componentName componentName -Db_info "DBHostName:Port:DBSvcName" \
   -Namespace "dc=domain"
```

- Adding an Oracle Internet Directory instance within an existing component of `Type=OID` by using `oidctl add`. See Creating and Starting an Oracle Internet Directory Server Instance by Using OIDCTL

The recommended method is to use `opmnctl` to add a system component. If you create an instance by adding a component with `opmnctl`, you must use `opmnctl` or Oracle Enterprise Manager Fusion Middleware Control, not `oidctl`, to stop and start the instance. See Section 8.3.7, "Starting the Oracle Internet Directory Server by Using opmnctl" and Section 8.2.2, "Starting the Oracle Internet Directory Server by Using Fusion Middleware Control."

You can update the configuration attributes of the instance by using Fusion Middleware Control, LDAP tools, or Oracle Directory Services Manager. See Chapter 9, "Managing System Configuration Attributes."

If you use `opmnctl` to add a system component with `oid2` as the component name, then an additional instance with `componentname=oid2` is configured within the given Oracle instance, which is `asinst_1` by default. This instance of Oracle Internet Directory can be started and stopped by using the `opmnctl` command with `ias-component=oid2` or by using Fusion Middleware Control. The instance-specific configuration entry for this instance is `cn=oid2,cn=osdldapd,cn=subconfigsubentry` and the configuration attributes in that entry can be updated to customize the instance. For more information about instance-specific configuration attributes, see Section 9.1.3, "Attributes of the Instance-Specific Configuration Entry."

> **Note:**
>
> You can use `oidctl` to create an instance if you are running Oracle Internet Directory as a standalone server, not part of a WebLogic domain. When you create an instance with `oidctl`, you must use `oidmon` and `oidctl` to stop and start the instance. An Oracle Internet Directory instance created with `oidctl` cannot be registered with a WebLogic server, so you cannot use Oracle Enterprise Manager Fusion Middleware Control to manage the instance. See Appendix B, "Managing Oracle Internet Directory Instances by Using OIDCTL."

11g Replication Server

Use `oidctl` or Oracle Enterprise Manager Fusion Middleware Control to start replication on an instance the first time. After that, `opmnctl` stops and starts replication when it stops and starts the component. If you must stop and start the Oracle Internet Directory Replication Server for administration purposes, use `oidctl` or Oracle Enterprise Manager Fusion Middleware Control.

11g OIDMON

In 11g Release 1, OIDMON monitors and reports the status of all Oracle Internet Directory processes (dispatcher, directory server, and replication server) to OPMN. This monitoring by OIDMON enables Fusion Middleware Control to report Oracle Internet Directory status accurately.

> **See Also:**
>
> - Understanding Process Control of Oracle Internet Directory Components
> - Managing Oracle Internet Directory Instances

A.1.1.2 Creating 12c Oracle Internet Directory Instance

Since 11g Release 1 (11.1.1.0.0), the procedure for creating an instance has changed. Configuration information for an Oracle Internet Directory instance resides in an instance-specific configuration entry, which has a DN of the form

`cn=componentname,cn=osdldapd,cn=subconfigsubentry`

where `componentname` is the name of a Oracle Fusion Middleware system component of `Type=OID`, for example, `oid1`. You do not manually create an instance-specific configuration entry. Instead, you create a Oracle Fusion Middleware component of `Type=OID`. Creating the Oracle Internet Directory component automatically generates an instance-specific configuration entry.

> **Note:**
>
> The entry in `configset0` is read-only and is used to store default attribute values for seeding new instance-specific configuration entries.

The first Oracle Internet Directory system component is created during installation. The first Oracle Internet Directory system component, `oid1` by default, is created during installation with the Oracle instance name `asinst_1` by default. The corresponding configuration entry for this component is `cn=oid1,cn=osdldapd,cn=subconfigsubentry`. There are two ways to create an additional Oracle Internet Directory instance:

- Adding another component of `Type=OID` by using `oid_createInstance`. For example:

  ```
  oid_createInstance(instanceName='instance-name', host='host', port='port')
  ```

  See Creating an Oracle Internet Directory Component by Using WLST Command — oid_createInstance for more information.
- Adding an Oracle Internet Directory instance within an existing component of `Type=OID` by using `oidctl add`. See Creating and Starting an Oracle Internet Directory Server Instance by Using OIDCTL for more information.

The recommended method is to use `oid_createInstance` to add a system component. If you create an instance by adding a component with WLST command, `oid_createInstance`, you must use `WLST` or Oracle Enterprise Manager Fusion Middleware Control, not `oidctl`, to stop and start the instance. See Starting the Oracle Internet Directory Server by Using WLST Command — start() and Starting the Oracle Internet Directory Server by Using Fusion Middleware Control .

You can update the configuration attributes of the instance by using Fusion Middleware Control, LDAP tools, or Oracle Directory Services Manager. See Managing System Configuration Attributes.

If you use `oid_createInstance` to add a system component with `oid2` as the component name, then an additional instance with `componentname=oid2` is configured within the given Oracle instance, which is `asinst_1` by default. This instance of Oracle Internet Directory can be started and stopped by using the `$DOMAIN_HOME/bin/startComponent.sh <instance-name>` or `$DOMAIN_HOME/bin/stopComponent.sh <instance-name>` respective command, with `ias-component=oid2` or by using Fusion Middleware Control. The instance-specific configuration entry for this instance is `cn=oid2,cn=osdldapd,cn=subconfigsubentry` and the configuration attributes in that entry can be updated to customize the instance. For more information about instance-specific configuration attributes, see Attributes of the Instance-Specific Configuration Entry.

> **Note:**
>
> You can use `oidctl` to create an instance if you are running Oracle Internet Directory as a standalone server, not part of a WebLogic domain. When you create an instance with `oidctl`, you must use `oidmon` and `oidctl` to stop and start the instance. An Oracle Internet Directory instance created with `oidctl` cannot be registered with a WebLogic server, so you cannot use Oracle Enterprise Manager Fusion Middleware Control to manage the instance. See Managing Oracle Internet Directory Instances by Using OIDCTL .

A.1.1.3 Starting and Stopping 11g Replication Server

Use `oidctl` or Oracle Enterprise Manager Fusion Middleware Control to start replication on an instance the first time. After that, Node Manager stops and starts replication when it stops and starts the component. If you must stop and start the Oracle Internet Directory Replication Server for administration purposes, use `oidctl` or Oracle Enterprise Manager Fusion Middleware Control.

A.1.1.4 About Monitoring and Reporting the Status of Oracle Internet Directory Processes to OPMN by 11g OIDMON

In 11g Release 1 (11.1.1.0.0), OIDMON monitors and reports the status of all Oracle Internet Directory processes (dispatcher, directory server, and replication server) to OPMN. This monitoring by OIDMON enables Fusion Middleware Control to report Oracle Internet Directory status accurately.

A.1.2 About Assigning SSL and non-SSL Ports

During installation of Oracle Internet Directory, Oracle Identity Management 11*g* Installer follows specific steps in assigning the SSL and non-SSL port. First, it attempts to use 3060 as the non-SSL port. If that port is unavailable, it tries ports in the range 3061 to 3070, then 13060 to 13070. Similarly, it attempts to use 3131 as its SSL port, then ports in the range 3132 to 3141, then 13131 to 13141.

If you want Oracle Internet Directory to use privileged ports, you can override the defaults during installation by using `staticports.ini`. (See *Installing and Configuring Oracle Internet Directory.)* You can also reset the port numbers after installation. See Enabling Oracle Internet Directory to run on Privileged Ports.

> **Note:**
>
> If you perform an upgrade from an earlier version of Oracle Internet Directory to 11g Release 1 (11.1.1.0.0), your port numbers from the earlier version are retained.

A.1.3 Changed Path Names in 12c Configuration and Log Files

In Oracle Fusion Middleware 12c Release 2, files that are updatable are installed under *DOMAIN_HOME* and most product binaries are stored under *ORACLE_HOME*. As a result, the path names of most configuration files and log files are different than in 11g.

Table A-1 lists some examples:

Table A-1 Some Path Names that Changed

Filename	11g Release 1 Location	12c Release 2 Location
Orclpwdlldap1 Oidpwdr*SID*	*ORACLE_INSTANCE*/OID/admin	*$DOMAIN_HOME*/config/fmwconfig/components/OID/admin
Tnsnames.ora	*ORACLE_HOME*/config	*$DOMAIN_HOME*/config/fmwconfig/components/OID/config
Oidldapd*.log oidmon*.log	*ORACLE_HOME*/diagnostics/logs/OID/*componentName*	*$DOMAIN_HOME*/servers/OID/logs/*componentInstance*
bulkload.log bulkdelte.log catalog.log	*ORACLE_HOME*/diagnostics/logs/OID/tools	*$DOMAIN_HOME*/tools/OID
Bulkload intermediate files	*ORACLE_INSTANCE*/OID/load	*$DOMAIN_HOME*/tools/OID/load
opmnctl	*ORACLE_INSTANCE/bin*	opmnctl Not supported in 12c
opmn.xmll	*ORACLE_INSTANCE*/config/OPMN/opmn	opmn Not supported in 12c
wlst.sh	Not applicable	*$ORACLE_HOME*/oracle_common/common/bin
startWeblogic.sh	*$DOMAIN_HOME*/bin	*$DOMAIN_HOME*/bin
startNodeManager.sh	Not applicable	*$DOMAIN_HOME*/bin
startComponent.sh	Not applicable	*$DOMAIN_HOME*/bin
stopComponent.sh	Not applicable	*$DOMAIN_HOME*/bin

> **See Also:**
>
> Understanding Oracle Internet Directory in Oracle Fusion Middleware

A.1.4 About Configuring Audit Framework Using Oracle Enterprise Manager Fusion Middleware Control

As of release 11g Release 1 (11.1.1.0.0), Oracle Internet Directory uses an audit framework that is integrated with Oracle Fusion Middleware. You can configure auditing by using Oracle Enterprise Manager Fusion Middleware Control or the WebLogic Scripting Tool, `wlst`.

The attribute `orclAudFilterPreset` can be set to `None`, `Low`, `Medium`, `All`, or `Custom`.

See Managing Auditing

A.1.5 Updated Server Chaining

Beginning from 11g, server chaining supports Novell eDirectory, as well as Microsoft Active Directory and Sun Java System Directory Server, formerly known as SunONE iPlanet.

The attributes `mapUIDtoADAttribute`, `showExternalGroupEntries`, `showExternalUserEntries`, and `addOrcluserv2ToADUsers` have been added since Oracle Internet Directory 10g (10.1.4.0.1).

A.1.6 About Setting Up and Managing LDAP-Based Replication

You can set up and manage LDAP-based replication by using the command line.

You can use LDAP-based replication for multimaster directory replication groups.

> **See Also:**
>
> - Understanding Oracle Internet Directory Replication
> - Advanced Administration: Directory Replication

A.1.7 About Java Containers

Oracle Directory Services Manager and Oracle Directory Integration Platform are Java components that run in WebLogic servers.

The Oracle Internet Directory LDAP and replication servers, as C programs, are system components and are not affected by this change. The Java server plug-ins run in a JVM within the `oidldapd` server itself. This is implemented using the Java Native Interface (JNI).

> **See Also:**
>
> - Product Overview in *Understanding Oracle WebLogic Server* guide
> - Developing Plug-ins for the Oracle Internet Directory Server

A.2 Managing Oracle Internet Directory Instances by Using OIDCTL

This appendix describes how to use OIDCTL to manage Oracle Internet Directory in standalone mode, without a WebLogic domain or Oracle Enterprise Manager Fusion Middleware Control. Specifically, this appendix describes how to use OIDCTL to create, delete, start, stop, and view status information for an Oracle Internet Directory server instance using OIDCTL.
This appendix includes the following sections:

- About Managing Oracle Internet Directory by Using OIDCTL
- Creating and Starting an Oracle Internet Directory Server Instance by Using OIDCTL
- About Stopping an Oracle Internet Directory Server Instance by Using OIDCTL
- About Starting an Oracle Internet Directory Server Instance by Using OIDCTL
- Viewing Status Information by Using OIDCTL
- Deleting an Oracle Internet Directory Server Instance by Using OIDCTL

> **See Also:**
>
> Starting an Instance of the Replication Server by Using OIDCTL.

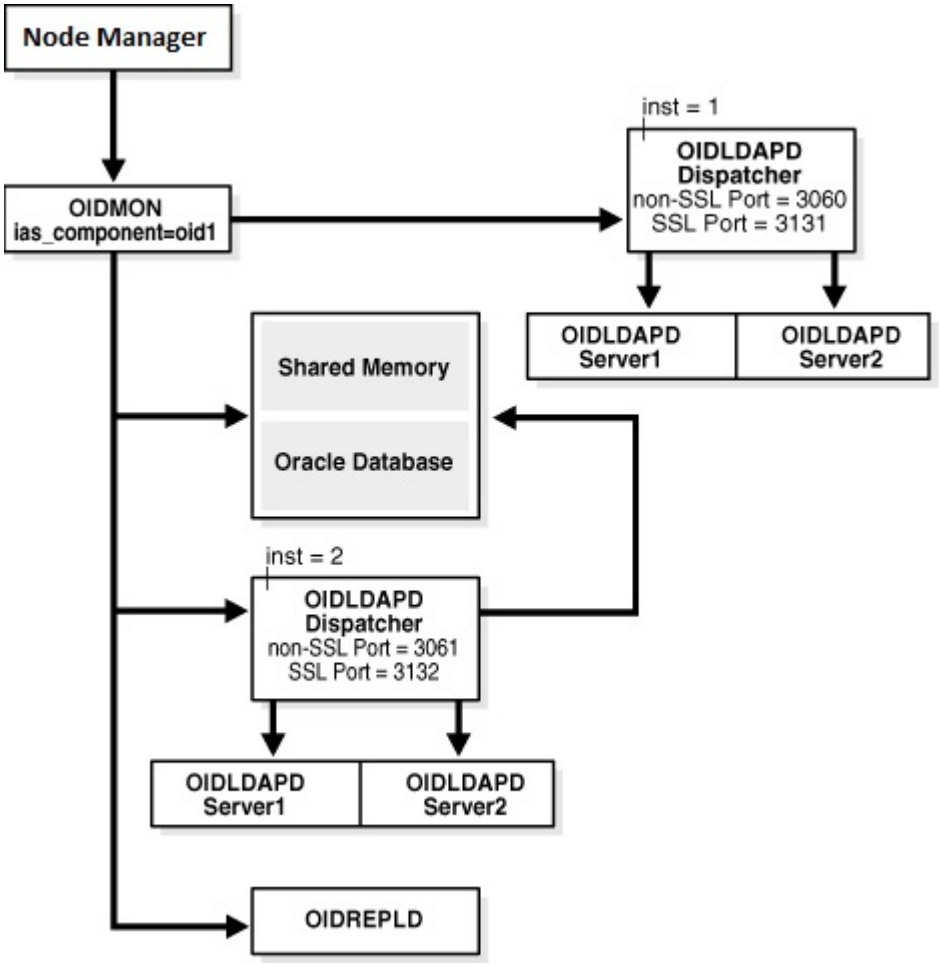
A.2.1 About Managing Oracle Internet Directory by Using OIDCTL

Some customers need to run Oracle Internet Directory in standalone mode, without a WebLogic domain or Oracle Enterprise Manager Fusion Middleware Control. If you are running Oracle Internet Directory in standalone mode, you can use `oidctl add` to create additional instances of Oracle Internet Directory within an existing Oracle Internet Directory component. If you create an Oracle Internet Directory instance by using `oidctl`, your new instance is not visible to a WebLogic server and you are not able to manage the instance by using Oracle Enterprise Manager Fusion Middleware Control.

The `inst` argument to `oidctl` is an integer. That is, the first server instance is `1` (`inst=1`) and the second is `2`, and so forth.

Figure A-1 shows a single Oracle Internet Directory component with two Oracle Internet Directory instances. Each instance has its own dispatcher with a non-SSL and an SSL port. Both dispatchers are controlled by the same OIDMON.

Figure A-1 A Component with Two Instances



Creating a new instance in this manner also creates a new instance-specific configuration entry with the new instance number appended to the component name. For example, if the first instance-specific configuration entry was:

```
cn=oid1,cn=osdldapd,cn=subconfigsubentry
```

then the second instance-specific configuration entry is:

```
cn=oid1_2,cn=osdldapd,cn=subconfigsubentry
```

You can manage the two instances independently by using LDAP tools or Oracle Directory Services Manager. The new instance cannot be registered with a WebLogic domain, so you cannot use Oracle Enterprise Manager Fusion Middleware Control or WLST to manage it.

Creating an Oracle Internet Directory instance in this manner does not generate any new pathnames in the file system. Instances that are part of the same Oracle Internet Directory component read from the same configuration files and write log files to the same log directory.

A.2.2 Creating and Starting an Oracle Internet Directory Server Instance by Using OIDCTL

You can create another Oracle Internet Directory server instance within an existing component and start the server by using `oidctl add`. This command starts the server as well.

> **Note:**
>
> - You must set the environment variables `DOMAIN_HOME` and `ORACLE_HOME` before you run the `oidctl` command.
> - If you do not specify the Oracle Internet Directory instance name and component name to `oidctl`, the command uses the default value `inst1` or `oid1` respectively. If you must specify different values, you can either set the environment variables `INSTANCE_NAME` and `COMPONENT_NAME`, or pass the instance name and component name in the command line as `name=`*`instanceName`*, `componentname=`*`componentName`*, respectively.
> - Best practice is to create new Oracle Internet Directory instances by creating new Oracle Internet Directory components, as described in Managing Oracle Internet Directory Instances. You should only use `oidctl` to create an instance if you plan to run Oracle Internet Directory in standalone mode and never use Oracle Enterprise Manager Fusion Middleware Control.

To create and start the server by using `oidctl add`, type:

```
oidctl connect=connect_string server=oidldapd inst=new_instance_number \
   name=instanceName componentname=componentName  \
   flags="port=non_ssl_port sport=ssl_port" add
```

`oidctl add` creates the new configuration entry:

```
 cn=componentName_new_instance_number,cn=osdldapd,cn=subconfigsubentry
```

Typically, the `inst` value of the original instance is `1`, the second instance you create is `2`, and so forth. For example:

```
oidctl connect=oiddb server=oidldapd inst=2 flags="port=3322 sport=3323" add
```

A.2.3 About Stopping an Oracle Internet Directory Server Instance by Using OIDCTL

If you have an Oracle Internet Directory server instance within a component that was created by using `oidctl`, and the instance has been started, you can stop it by executing the following command.

To stop the server instance, execute the following command:

```
oidctl connect=connect_string server=oidldapd inst=new_instance_number stop
```

For example:

```
oidctl connect=oiddb server=oidldapd inst=2 stop
```

A.2.4 About Starting an Oracle Internet Directory Server Instance by Using OIDCTL

If you have an Oracle Internet Directory server instance within a component that was created by using `oidctl`, and the instance has been stopped, you can start it by executing the following command.

To start the server instance, execute the following command:

```
oidctl connect=connect_string server=oidldapd inst=new_instance_number start
```

For example:

```
oidctl connect=oiddb server=oidldapd inst=2 start
```

The server is started with the port and sport flags that were specified when the instance was created.

A.2.5 Viewing Status Information by Using OIDCTL

You can use `oidctl` to view status.

Type:

```
oidctl connect=connect_string status
```

The command `oidctl status` displays the status of all the Oracle Internet Directory instances that are running on the host, even though they might be on different Oracle instances.

A.2.6 Deleting an Oracle Internet Directory Server Instance by Using OIDCTL

You can stop and permanently delete one Oracle Internet Directory server instance within a component by executing the command specified below.

To stop and permanently delete one Oracle Internet Directory server instance within a component, type

```
oidctl connect=connect_string server=oidldapd inst=new_instance_number \
   name=instanceName componentname=componentName delete
```

Only the `connect`, `server`, and `inst` arguments are required. `oidctl delete` deletes the configuration entry:

```
cn=componentName_new_instance_number,cn=osdldapd,cn=subconfigsubentry
```

Typically, the `inst` value of the original instance is `1`, the second instance you create is `2`, and so forth. For example:

```
oidctl connect=oiddb server=oidldapd inst=2 delete
```

A.3 How Replication Works

This appendix describes how replication works, including the features and architecture of LDAP-based replication, LDAP replica states, and the replication process.
This appendix includes the following sections:

- Architecture of LDAP-Based Replication
- LDAP Replica States
- Managing an Entry Using Multimaster Replication Process

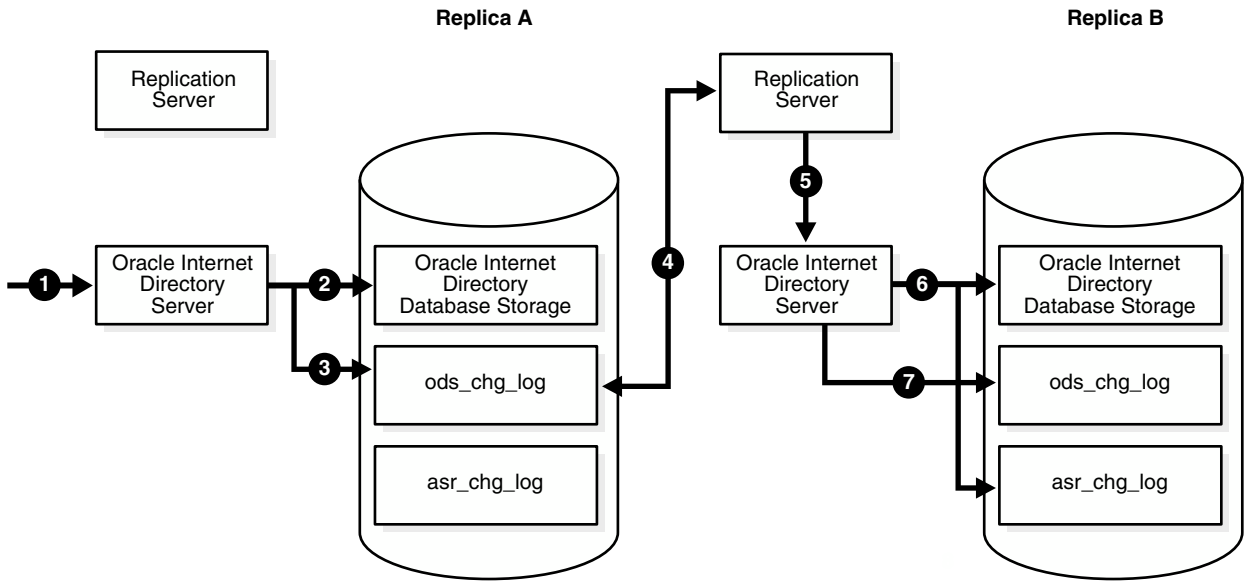
> **Note:**
>
> All references to Oracle Single Sign-On in this appendix refer to Oracle Single Sign-On 10g (10.1.4.3.0) or later.

A.3.1 Architecture of LDAP-Based Replication

The data flow in LDAP replication consists of the apply phase with the apply queue. The transport phase with the transport queue is no longer used. This section gives a more detailed look at LDAP replication.

Figure A-2 and its accompanying text explain the fan-out replication process.

Figure A-2 LDAP Replication Process



1. An LDAP change request is made on the Oracle Internet Directory server of Replica A.
2. The change is accepted and committed to Oracle Internet Directory database storage.
3. A new change log is generated for the operation and stored in the `ods_chg_log` table, where:

- server = Replica A
- orclreplicationid/chg_rid = 0 (The value 0 is for user operations.)
- Modifiersname = user DN of the Modifier (by default)

4. The Replication Server at Replica B fetches the change logs from the ods_chg_log table at Replica A, performs some partial replication filtering in memory, and then puts the change logs directly into the apply queue. The orclupdateschedule configuration attribute determines the replication frequency.
5. The Replication Server at Replica B requests that the Oracle Internet Directory server at Replica B commit the changes to storage.
6. The Oracle Internet Directory server at Replica B commits the changes to the Oracle Internet Directory database storage, copies the changes into the asr_chg_log table, and updates the retry count to -2, all in a single LDAP call.

   If a failure occurs on the commit, the Oracle Internet Directory server reports the error to the Replication Server, copies the changes to the asr_chg_log table, and decrements the retry count by 1. The Oracle Internet Directory server can then re-apply the commit.

   Regardless of success or failure, all change logs are be copied to the asr_chg_log table at Replica B.
7. If Replica B is the source for any other replicas, a change log is regenerated in ods_chg_log table at Replica B, which follows the following changelog regeneration rules:

   - server = server of local replica, Replica B
   - orclreplicationid/chg_rid = N
   - Modifiersname = Replbind_DN_of_A

   orclreplicatid/chg_rid is set to the orclreplicationid value of the processing agreement.

A.3.2 LDAP Replica States

Understand about the replica states for LDAP-based replication.

When LDAP based replication is configured, and the replication server is started, the server reads the replica state orclreplicastate from the local replica, "orclreplicaid=*local_Replica_ID*, cn=replication configuration". The replication server behaves differently, based upon the local replica state, as shown in Table A-2. The replication server reads the local replica state from the local (consumer) node.

> **Note:**
>
> On Windows systems, ensure that the replication server is not running before you enable bootstrapping by changing the value of orclReplicaState to 0.

Table A-2 LDAP Replica States

Value	Meaning	Server Behavior
0	Bootstrap	Starts replication bootstrap processing to synchronize the consumer directory from the supplier, based on the replication naming context configuration. Updates the replica state to correspond with bootstrap progress. • Sets the replica state to 3 (Bootstrap in progress) immediately when it starts to bootstrap. • Sets the replica state to 4 (Bootstrap in progress, `cn=oraclecontext` bootstrap has completed) after it completes bootstrapping of `"cn=oraclecontext"` successfully • Sets the replica state to 5 (Bootstrap Error occurred) after bootstrap is completed but failure(s) detected during the bootstrap. Then it waits until the replica state is re-set. Note: Human intervention is required; See Troubleshooting Oracle Internet Directory Replication for details. • Sets the replica state to 1 (On-line) after bootstrap has completed successfully. Then replication automatically starts to perform normal replication processing.
1	On line	Starts normal replication processing to replicate changes from the supplier to the consumer.
2	Off line	Logs an error message in oidrepld.log similar to this: `2004/09/24:17:41:44 * Replica(dlsun1418_replica2) is in OFFLINE mode, Please update the replica state and restart OIDREPLD...` The administrator must set the replica state properly and restart the replication server.
3	Bootstrap in progress	Sets the replica state back to 0 (Bootstrap), then starts to bootstrap again as if the replica state were 0.
4	Bootstrap in progress, `cn=oraclecontext` bootstrap has completed.	Sets the replica state back to 0 (Bootstrap), then starts to bootstrap again as if the replica state were 0.
5	Bootstrap completed; failure detected for one or more naming contexts.	Logs an error message in oidrepld.log similar to this: `2004/09/24:17:13:30 * Replication BOOTSTRAP_ERROR mode detected for replica(dlsun1418_replica2)` Then it waits until the replica state is reset properly.
6	Database copy-Based addnode;	This mode indicates that the replica is a database copy-based addnode.
7	Sync schema	Sync schema, but not data, from supplier to consumer.
8	Boot strap without schema sync	If schema sync was previously performed, but bootstrapping failed at a subsequent step, bootstrapping is started again without performing another schema sync.

Oracle Internet Directory replication server logs the bootstrapping process in the Oracle Internet Directory replication server log, *$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidrepld00.log`.

If bootstrap completes successfully, the log looks similar to the following example, and the replication server automatically starts to perform normal replication processing.

```
2004/10/06:17:13:25 * Starting OIDREPLD against isunnad03:5555...
2004/10/06:17:13:26 * Starting scheduler...
2004/10/06:17:13:27 * Start to BootStrap from supplier=isunnad03_purify to
consumer=isunnad03_purify3
2004/10/06:17:13:28 * gslrbssSyncDIT:Replicating namingcontext=cn=oraclecontext ......
2004/10/06:17:14:21 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oraclecontext, 222 entries matched
2004/10/06:17:14:21 * gslrbssSyncDIT:Replicating namingcontext=c=india ......
2004/10/06:17:14:21 * gslrbssSyncDIT:Sync done successfully for namingctx: c=india, 0
entries matched
2004/10/06:17:14:21 * gslrbssSyncDIT:Replicating namingcontext=c=uk ......
2004/10/06:17:19:57 * gslrbssSyncDIT:Sync done successfully for namingctx: c=uk, 1087
entries matched
2004/10/06:17:19:57 * gslrbssSyncDIT:Replicating
namingcontext=cn=oracleschemaversion ......
2004/10/06:17:19:59 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oracleschemaversion, 10 entries matched
2004/10/06:17:20:01 * gslrbsbBootStrap: BOOTSTRAP DONE SUCCESSFULL
```

If failures are detected, the log looks similar to the following example:

```
2004/09/14:12:57:23 * Starting OIDREPLD against dlsun1418:4444...
2004/09/14:12:57:25 * Starting scheduler...
2004/09/14:12:57:26 * Start to BootStrap from supplier=dlsun1418_replica to
consumer=dlsun1418_replica2
2004/09/14:12:57:27 * gslrbssSyncDIT:Replicating namingcontext=cn=oraclecontext ......
2004/09/14:12:58:21 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oraclecontext, 222 entries matched
2004/09/14:12:58:21 * gslrbssSyncDIT:Replicating namingcontext=cn=quan zhou ......
2004/09/14:12:58:23 * BootStrap failure when adding DN=cn=Quan
Zhou,server=dlsun1418_replica2,err=Constraint violation.
2004/09/14:12:58:23 * gslrbssSyncDIT:Sync failed for namingctx: cn=quan zhou, only 1
entries retrieved
2004/09/14:12:58:23 * gslrbssSyncDIT:Replicating
namingcontext=cn=oracleschemaversion ......
2004/09/14:12:58:25 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oracleschemaversion, 10 entries matched
2004/09/14:12:58:51 * gslrbsbBootStrap: Failure occurred when bootstrapping 1 out of 3
namingcontext(s) from the supplier
```

Tip:

You have two options for troubleshooting bootstrap failure.

- Option 1: Identify the cause of the bootstrap failure and fix the cause, then restart bootstrapping by setting the consumer replica's `orclreplicastate` to Bootstrap mode.
- Option 2: Identify the naming contexts that failed to be bootstrapped and use `oidcmprec` to reconcile them. Then resume replication by setting the consumer replica's `orclreplicastate` to Online mode.

> **Note:**
>
> `Oidrepld` is now in Bootstrap_error mode, so you do need to reset the consumer replica's replica state (`orclreplicastate`).

A.3.3 Managing an Entry Using Multimaster Replication Process

This section describes how the multimaster replication process adds, deletes, and modifies entries, and how it modifies DNs and RDNs. It contains these topics:

- How the Multimaster Replication Process Adds a New Entry to a Consumer
- How the Multimaster Replication Process Deletes an Entry
- How the Multimaster Replication Process Modifies an Entry
- How the Multimaster Replication Process Modifies a Relative Distinguished Name
- How the Multimaster Replication Process Modifies a Distinguished Name

A.3.3.1 How the Multimaster Replication Process Adds a New Entry to a Consumer

When the directory replication server adds a new entry to a consumer, it follows this change application process:

1. The directory replication server looks in the consumer for the DN of the parent of the target entry. Specifically, it does this by looking for a global unique identifier (GUID) assigned to the DN of the parent.
2. If the parent entry exists, then the directory replication server composes a DN for the new entry and places the new entry under its parent in the consumer. It then places the change entry in the purge queue.

If the change entry is not successfully applied on the first try, then:

The directory replication server places the new change entry in the retry queue, sets the number of retries to the configured maximum, and repeats the change application process.

If the change entry is not successfully applied on *all but the last retry*, then:

The directory replication server keeps the change entry in the retry queue, decrements the number of retries, and repeats the change application process.

If the change entry is not successfully applied on the last retry, then:

The directory replication server checks to see if the new entry is a duplicate of an existing entry.

If the change entry is a duplicate entry, then:

The directory replication server applies the following conflict resolution rules:

- The entry with the older creation time stamp is used.
- If both entries have the same creation time stamp, then the entry with the smaller GUID is used.

If the change entry is used, then the target entry is removed, the change is applied, and the change entry is placed in the purge queue.

If the target entry is used, then the change entry is placed in the purge queue.

If the change entry is not a duplicate entry, then:

The directory replication server places the change entry in the human intervention queue, and repeats the change application process at the interval you specified in the `orclHIQSchedule` parameter.

If the change entry is not successfully applied after it has been placed in the human intervention queue:

The directory replication server keeps the change in this queue, and repeats the change application process at specified intervals while awaiting action by the administrator. The administrator can use the Oracle Internet Directory Comparison and Reconciliation Tool and the human intervention queue manipulation tool to resolve the conflict.

A.3.3.2 How the Multimaster Replication Process Deletes an Entry

When the directory replication server deletes an entry from a consumer, it follows this change application process:

1. The directory replication server looks in the consumer for an entry with a GUID matching the one in the change entry.
2. If the matching entry exists in the consumer, then the directory replication server deletes it. It then places the change entry in the purge queue.

If the change entry is not successfully applied on the first try, then:

The directory replication server places the change entry in the retry queue, sets the number of retries to the configured maximum, and repeats the change application process.

If the change entry is not successfully applied on *all but the last retry*, then:

The directory replication server keeps the change entry in the retry queue, decrements the number of retries, and repeats the change application process.

If the change entry is not successfully applied on the last retry, then:

The directory replication server places the change entry in the human intervention queue and repeats the change application process at specified intervals.

If the change entry is not successfully applied after it has been placed in the human intervention queue:

The directory replication server keeps the change entry in this queue, and repeats the change application process at specified intervals while awaiting action by the administrator. The administrator can use the Oracle Internet Directory Comparison and Reconciliation Tool and the human intervention queue manipulation tool to resolve the conflict.

A.3.3.3 How the Multimaster Replication Process Modifies an Entry

When the directory replication server modifies an entry in a consumer, it follows this change application process:

1. The directory replication server looks in the consumer for an entry with a GUID matching the one in the change entry.
2. If the matching entry exists in the consumer, then the directory replication server compares each attribute in the change entry with each attribute in the target entry.
3. The directory replication server then applies the following conflict resolution rules:

a. The attribute with the most recent modify time is used.

b. The attribute with the most recent version of the attribute is used—for example, version 1, 2, or 3.

c. The modified attribute on the host whose name is closest to the beginning of the alphabet is used.

4. The directory replication server applies the filtered modification, and places the change entry in the purge queue.

If the change entry is not successfully applied on the first try, then:

The directory replication server places the change entry in the retry queue, sets the number of retries to the configured maximum, and repeats the change application process.

If the change entry is not successfully applied on *all but the last retry*, then:

The directory replication server keeps the change entry in the retry queue, decrements the number of retries, and repeats the change application process.

If the change entry is not successfully applied by the last retry, then:

The directory replication server places the change entry in the human intervention queue and repeats the change application process at specified intervals.

If the change entry is not successfully applied after it has been placed in the human intervention queue:

The directory replication server keeps the change entry in this queue, and repeats the change application process at specified intervals while awaiting action by the administrator. You can use the Oracle Internet Directory Comparison and Reconciliation Tool and the Human Intervention Queue Manipulation Tool to resolve the conflict.

A.3.3.4 How the Multimaster Replication Process Modifies a Relative Distinguished Name

When the directory replication server modifies the RDN of an entry in a consumer, it follows this change application process:

1. The directory replication server looks in the consumer for the DN with a GUID that matches the GUID in the change entry.
2. If the matching entry exists in the consumer, then the directory replication server modifies the RDN of that entry and places the change entry in the purge queue.

If the change entry is not successfully applied on the first try, then:

The directory replication server places the change entry in the retry queue, sets the number of retries to the configured maximum, and repeats the change application process.

If the change entry is not successfully applied on *all but the last* retry, then:

The directory replication server keeps the change entry in the retry queue, decrements the number of retries, and repeats the change application process.

If the change entry is not successfully applied on the last retry, then:

The directory replication server places the change entry in the human intervention queue and checks to see if it is a duplicate of the target entry.

If the change entry is a duplicate entry, then:

The directory replication server applies the following conflict resolution rules:

- The entry with the older creation time stamp is used.
- If both entries have the same creation time stamp, then the entry with the smaller GUID is used.

If the change entry is used, then the target entry is removed, the change entry is applied, and then placed in the purge queue.

If the target entry is used, then the change entry is placed in the purge queue.

If the change entry is not a duplicate entry, then:

The directory replication server places the change entry in the human intervention queue, and repeats the change application process at specified intervals.

If the change entry is not successfully applied after it has been placed in the human intervention queue:

The directory replication server keeps the change entry in this queue, and repeats the change application process at specified intervals while awaiting action by the administrator. The administrator can use the Oracle Internet Directory Comparison and Reconciliation Tool and the Human Intervention Queue Manipulation Tool to resolve the conflict.

A.3.3.5 How the Multimaster Replication Process Modifies a Distinguished Name

When the directory replication server modifies the DN of an entry in a consumer, it follows this change application process:

1. The directory replication server looks in the consumer for the DN with a GUID that matches the GUID in the change entry.

   The directory replication server also looks in the consumer for the parent DN with a GUID that matches the GUID of the new parent specified in the change entry.
2. If both the DN and the parent DN of the target entry exist in the consumer, then the directory replication server modifies the DN of that entry and places the change entry in the purge queue.

If the change entry is not successfully applied on the first try, then:

The directory replication server places the change entry in the retry queue, sets the number of retries to the configured maximum, and repeats the change application process.

If the change entry is not successfully applied on *all but the last retry*, then:

The directory replication server keeps the change entry in the retry queue, decrements the number of retries, and repeats the change application process.

If the change entry is not successfully applied by the last retry, then:

The directory replication server places the change entry in the human intervention queue and checks to see if it is a duplicate of the target entry.

If the change entry is a duplicate entry, then:

The directory replication server applies the following conflict resolution rules:

- The entry with the older creation time stamp is used.
- If both entries have the same creation time stamp, then the entry with the smaller GUID is used.

If the change entry is used, then the target entry is removed, the change entry is applied, and then placed in the purge queue.

If the target entry is used, then the change entry is placed in the purge queue.

If the change entry is not a duplicate entry, then:

The directory replication server places the change entry in the human intervention queue, and repeats the change application process at specified intervals.

If the change entry is not successfully applied after it has been placed in the human intervention queue:

The directory replication server keeps the change entry in this queue, and repeats the change application process at specified intervals while awaiting action by the administrator. The administrator can use the Oracle Internet Directory Comparison and Reconciliation Tool and the Human Intervention Queue Manipulation Tool to resolve the conflict.

A.4 Java Server Plug-in Developer's Reference

This appendix describes the Java API in the server plug-in framework for Oracle Internet Directory.
In response to both customer and internal requests, Oracle has added a Java API to the server plug-in framework for Oracle Internet Directory. Some of the new Oracle Internet Directory features, such as server chaining, were developed using the Java plug-in API.

This appendix includes the following sections:

- Advantages of Java Plug-ins
- Setting Up a Java Plug-in
- Overview of Java Plug-in API
- Java Plug-in Error and Exception Handling Examples
- Java Plug-in Debugging and Logging
- Java Plug-in Examples

A.4.1 Advantages of Java Plug-ins

In addition to the advantages of the Java language itself, Java server plug-ins offer many advantages over PL/SQL plug-ins

They are:

- Bidirectional communication between the server and the plug-in
- The ability of the plug-in to return a search result
- Support for the `moddn` operation
- Better performance
- No knowledge of database required
- Enhanced security
- Enhanced debugging capability

A.4.2 Setting Up a Java Plug-in

You can set up a Java plug-in as described in the topic below.

To set up a Java Plug-in:

1. Create the standalone Java program using the pre-defined class definition and methods. You can implement the plug-in as a jar file or as a package.
2. Compile the plug-in file or package. Before compiling, ensure that your CLASSPATH is set to `$ORACLE_HOME/ldap/jlib/ospf.jar`. Make sure the compilation completes without error.
3. Place the class file, jar, or package in the pre-defined class location `$ORACLE_HOME/ldap/server/plugin`.
4. Register the Java plug-in by adding the plug-in configuration entry.

   You can add the entry by using the command line or by using Oracle Directory Services Manager. For details, see Registering a Plug-in From the Command Line and Managing Plug-ins by Using Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control.

The jar file can have any name. The manifest file must contain the attribute Main-Class, followed by the name of the Java plug-in. For example:

```
Main-Class: myjavaplugin
```

The value of the `orclPluginName` attribute in the plug-in configuration entry must correspond with one of the following:

- The name of a class in a class file
- The fully qualified name of a class in a package
- A jar file name.

Table A-3 lists the Plug-In Names and the corresponding paths.

After you perform these steps, the server invokes the plug-in whenever the invocation criteria are met.

The classes included in the jar file must not occur in the environment. If they do, unexpected errors might occur. To correct this problem, remove the classes from the environment and restart the Oracle Internet Directory server. If the JAR or class file depends on other JAR files or class files, then append the dependent JAR files or paths of the class files to the `CLASSPATH` and restart the Oracle Internet Directory server.

You can control whether the server reloads the Java plug-in class every time the plug-in executes. If the value of the attribute `orclPluginClassReloadEnabled` is 1, the server reloads the plug-in class every time. If it is 0, the server loads the class only the first time the plug-in executes.

The path of the Oracle Internet Directory Server Plug-in Framework jar file is `$ORACLE_HOME/ldap/jlib/ospf.jar`.

A.4.3 orclPluginName Value

The value of the `orclPluginName` attribute determines where the server expects to find the class or jar file.

Table A-3 shows some examples.

Table A-3 Plug-in Names and Corresponding Paths

orclPluginName	Path
myjavaplugin	*ORACLE_HOME*/ldap/server/plugin/myjavaplugin.class
myjavaplugin.jar	*ORACLE_HOME*/ldap/server/plugin/myjavaplugin.jar
my.package.myjavaplugin	*ORACLE_HOME*/ldap/server/plugin/my/package/myjavaplugin

A.4.4 Overview of Java Plug-in API

You can get a high-level overview of the API and understand the role of the main classes and interfaces from this section.

For detailed information about all the Java server plug-in classes and interfaces, See Oracle® Fusion Middleware Java API Reference for Oracle Internet Directory.

This sections contains the following topics:

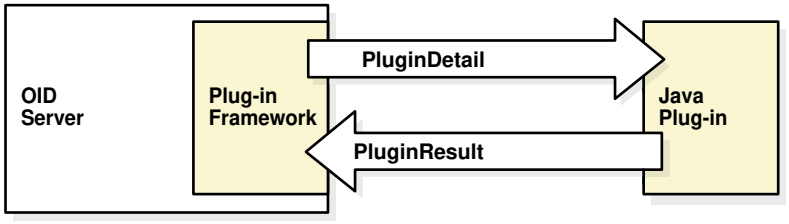
- Communication Between the Server and Plug-in
- Java Plug-in Structure
- Overview of PluginDetail
- PluginResult
- ServerPlugin Interface Methods for LDAP Operations

> **Note:**
>
> Do not use System.exit() in a Java plug-in. Doing so might lead to unpredictable behavior by the Oracle directory server.

A.4.4.1 Communication Between the Server and Plug-in

All Java plug-ins use the `ServerPlugin` interface for communication between the plug-in and the Oracle Internet Directory server. When the server invokes a Java plug-in, it constructs a `PluginDetail` object and passes information to the plug-in for that object. The plug-in constructs a `PluginResult` object. After it completes its task, the plug-in passes the `PluginResult` object back to the server. In some cases, the plug-in changes or adds to the information it received in the `PluginDetail` and passes the information back to the server in the `PluginResult` object. Figure A-3 shows the communication between the Oracle Internet Directory server and the Java Plug-ins.

Figure A-3 Communication Between the Server and the Java Plug-in



The Java plug-in can also use a `ServerLog` class to log messages in a log file.

A.4.4.2 Java Plug-in Structure

The general structure for a Java plug-in is:

```
public class Java_Plug-in_Class_Name {extends ServerPluginAdapter} {
         public PluginResult
Name_of_ServerPlugin_Method(PluginDetail plgObj)
throws Exception {
      // Plug-in Code
      }
}
```

or

```
public class Java_Plug-in_Class_Name {implements ServerPlugin}  {
         public PluginResult
Name_of_ServerPlugin_Method(PluginDetail plgObj)
throws Exception {
      // Plug-in Code
      }
}
```

A.4.4.3 Overview of PluginDetail

The `PluginDetail` contains the information described in the following sections:

- Server Object
- LdapBaseEntry
- LdapOperation Behavior
- PluginFlexfield

A.4.4.3.1 Server Object

This object contains metadata information about the Oracle Internet Directory Server where the plug-in is being executed. It contains the following information:

- Hostname
- Port
- `LdapContext`

The Hostname and the Port indicate the host and port on which the server is running.

The `LdapContext` object allows the plug-in to connect back to the server and inform it that the connection is being acquired from the plug-in. This is necessary, for example, in an ldapbind plug-in that performs an ldapbind itself. By connecting back to the server using this `LdapContext` object, the plug-in prevents the server from invoking the same plug-in, resulting in an infinite loop.

The following code fragment shows how the plug-in retrieves the `Server` object from the `PluginDetail` and connects back to the server:

```
// An LDAP Bind Plug-in
public class MyBindPlugin extends ServerPluginAdapter
{
  …..
```

```
  // Retrieve the Server Object from the PluginDetail
  Server srvObj = plgObj.getServer();
  ……
  // This bind will not result in the LDAP Bind Plug-in being called
  // in an infinite loop
  InitialLdapContext myConn =

(InitialLdapContext)srvObj.getLdapContextFromServerPlugin();
  myConn.bind(…);
  ….
}
```

See Oracle® Fusion Middleware Java API Reference for Oracle Internet Directory for information about the methods used in the example.

A.4.4.3.2 LdapBaseEntry

The `LdapBaseEntry` contains the following information:

- DN
- Attributes

The server must send DN information for all of the operations, except ldapadd. The meaning of the DN for each operation is shown in Table A-4.

Table A-4 The Meaning of the DN Information for Each LDAP Operation

Operation	Meaning of DN
ldapadd	No DN sent
ldapbind	The entry to which the directory server is attempting to bind
ldapcompare	The base entry on which to perform the compare
ldapdelete	For Pre and When timings, the entry which is to be deleted For Post timing, no DN sent
ldapmoddn	The base entry to be moved
ldapmodify	For Pre and When timings, the entry on which the modification is being performed For Post timing, the modified entry
ldapsearch	The base entry for the search

The Attributes are JNDI attributes.

The `LdapBaseEntry` has methods for accessing the DN and Attributes. For performance reasons, if the `LdapBaseEntry` is a group entry, and the entry cache capability is disabled, the attributes `uniquemember` and `member` are not accessible.

> **See Also:**
>
> Oracle Internet Directory Performance Tuning in *Tuning Performance*.

A.4.4.3.3 LdapOperation Behavior

This section explains what LdapOperation is and the various operations such as, add, bind, delete, compare, search, moddn, and modify.

This section contains the following topics:

- LDAP Operation
- AddLdapOperation
- BindLdapOperation
- CompareLdapOperation
- DeleteLdapOperation
- ModdnLdapOperation
- ModifyLdapOperation
- SearchLdapOperation

A.4.4.3.3.1 LDAP Operation

Every plug-in is associated with one of the seven basic LDAP operations: add, bind, compare, delete, moddn, modify, or search. The `LdapOperation` object contains the following information, which is passed to all seven operations:

- Bind DN
- Server Controls
- Operation Result Code

The Bind DN is the DN of the identity that is requesting the LDAP operation. Server Controls is a vector that contains control information. If any server controls are passed to the server during an operation, then the control information is passed to the Java plug-in for the Server Controls. The meaning of the Operation Result Code depends on the timing of the operation, as shown in Table A-4. Note that in the case of a when_replace operation, the plug-in can change the information in the Operation Result Code and pass it to the server in the `PluginResult`.

Table A-5 Behavior of Operation Result Code

Plug-in Timing	Meaning and Behavior of Operation Result Code
Pre When	Not used
When_replace	Error status of the LDAP operation performed by the plug-in. Output from the plug-in to the server.
Post	Error status of LDAP operation performed by the server. Input to the plug-in from the server.

`LdapOperation` also has methods for retrieving and modifying its contents.

Seven different classes representing the seven LDAP operations extend the `LdapOperation` class. Each of the subclasses includes class-specific information, in addition to the `LdapOperation` information. The classes and class-specific information are shown in Table A-6. Each class name in Table A-6 is a link to the section describing the details of that class:

Table A-6 Subclasses of LdapOperation and Class-specific information.

Class	Class-Specific information
AddLdapOperation	`LdapEntry`
BindLdapOperation	Bind Password
CompareLdapOperation	Attribute Name Attribute Value
DeleteLdapOperation	Delete DN
ModdnLdapOperation	New Parent DN New Relative DN Delete Old RDN New DN
ModifyLdapOperation	`LdapModification`
SearchLdapOperation	Filter Required Attributes Scope `SearchResultSet` (Not sent by server; created by plug-in to return data)

Each class has methods for creating, modifying, and retrieving its information. The class-specific information represents either input to the plug-in, output from the plug-in to the server, or both.

The rest of this section discusses the operation-specific classes in detail.

A.4.4.3.3.2 AddLdapOperation

When invoking an ldapadd plug-in, the server constructs an `AddLdapOperation` object containing a `LdapEntry` object to pass information about the entry that is being added. The LdapEntry Object contains the following information:

- DN
- Attributes

The DN represents the DN of the entry to be added. The Attributes are the entry's JNDI Attributes. As Table A-7 shows, for all operations except the post-operation, the plug-in can modify the information in the `LdapEntry` and return it to the server.

Table A-7 Behavior of LdapEntry Information for Each Plug-in Timing

Plug-in Timing	Behavior of LdapEntry Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

A.4.4.3.3.3 BindLdapOperation

The server passes the following information to an ldapbind plug-in:

- Bind Password
- Proxy Requester DN

Bind Password is the password for the bind. Proxy Requester DN is the DN of the identity requesting a Proxy Switch.

A.4.4.3.3.4 CompareLdapOperation

The server passes the following information to an ldapcompare plug-in:

- Attribute Name
- Attribute Value

The Attribute Name is the name to be compared during the ldapcompare operation. As Table A-8 shows, for all operations except the post-operation, the plug-in can modify the information in the Attribute Name and return it to the server.

Table A-8 Behavior of the AttributeName for Each Plug-in Timing

Plug-in Timing	Behavior of the Attribute Name Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The Attribute Value is the value to be compared during the ldapcompare operation. As Table A-9 shows, for all operations except the post-operation, the plug-in can modify the information in the Attribute Value and return it to the server.

Table A-9 Behavior of the Attribute Value for Each Plug-in Timing

Plug-in Timing	Behavior of the Attribute Value Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

A.4.4.3.3.5 DeleteLdapOperation

The server passes the Delete DN object to an ldapdelete plug-in. This is the DN to be deleted. As Table A-10 shows, for all operations except the post-operation, the plug-in can modify the information in the Delete DN and return it to the server.

Table A-10 Behavior of the Delete DN for Each Plug-in Timing

Plug-in Timing	Behavior of the DeleteDN Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

A.4.4.3.3.6 ModdnLdapOperation

The server passes the following information to ldapmoddn plug-ins:

- New Parent DN
- New Relative DN
- Delete Old RDN
- New DN

The New Parent DN contains the new parent of the RDN that was specified in the `LdapBaseEntry` of the `PluginDetail`. As Table A-11 shows, for all operations except the post-operation, the plug-in can modify the information in the New Parent DN and return it to the server.

Table A-11 Behavior of New Parent DN Information for Each Plug-in Timing

Plug-in Timing	Behavior of the New Parent DN Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The New Relative DN is the new RDN that is to replace the RDN that was specified in the `LdapBaseEntry` of the `PluginDetail`. As Table A-12 shows, for all operations except the post-operation, the plug-in can modify the information in the New Relative DN and return it to the server.

Table A-12 Behavior of New Relative DN Information for Each Plug-in Timing

Plug-in Timing	Behavior of the New Relative DN Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The Delete Old RDN value specifies whether the old RDN specified in the `LdapBaseEntry` of the `PluginDetail` is to be retained after it is replaced by the new relative DN. As Table A-13 shows, for all operations except the post-operation, the plug-in can modify the value in Delete Old RDN and return it to the server.

Table A-13 Behavior of Delete Old RDN Information for Each Plug-in Timing

Plug-in Timing	Behavior of the Delete Old RDN Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The New DN specifies the target DN in of the ldapmoddn operation. This information is only an input from the server to the plug-in. The plug-in cannot modify this information and return it to the server.

A.4.4.3.3.7 ModifyLdapOperation

The server passes an `LdapModification` object to ldapmodify plug-ins. The `LdapModification` object contains Modification Items, which are JNDI modification items. As Table A-14 shows, for all operations except the post-operation, the plug-in can modify the information in the `LdapModification` and return it to the server.

Table A-14 Behavior of LdapModification Information for Each Plug-in Timing

Plug-in Timing	Behavior of the LdapModification Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

A.4.4.3.3.8 SearchLdapOperation

The `SearchLdapOperation` object contains the following information:

- Filter
- Required Attributes
- Scope
- `SearchResultSet`

The Filter, Required Attributes, and Scope are passed by the server.

The Filter contains the LDAP search filter specified for the ldapsearch operation. This is only an input to the plug-in. The plug-in cannot modify this information and return it to the server.

The Required Attributes contains the required attributes specified for the ldapsearch operation. As Table A-15 shows, for all operations except the post-operation, the plug-in can modify the information in the Required Attributes and return it to the server.

Table A-15 Behavior of the Required Attributes for Each Plug-in Timing

Plug-in Timing	Behavior of the Required Attributes Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The Scope contains the scope of the search to be performed by the ldapsearch operation. As Table A-16 shows, for all operations except the post-operation, the plug-in can modify the information in the Scope and return it to the server.

Table A-16 Behavior of the Scope for Each Plug-in Timing

Plug-in Timing	Behavior of the Scope Information
Pre When When_replace	Both input and output. The plug-in can modify the information and return it to the server.
Post	Input only.

The `SearchResultSet` defines search results returned from the Java plug-in to the server. A plug-in performing an ldapsearch operation can construct this object. As Table A-17 shows, only the when and when_replace plug-ins can return a SearchResult Set to the server.

Table A-17 Behavior of the SearchResultSet for Each Plug-in Timing

Plug-in Timing	Behavior of the SearchResultSet Information
Pre	The plug-in cannot return the object.
When When_replace	The plug-in can return this object to the server.
Post	The plug-in cannot return the object.

A.4.4.3.4 PluginFlexfield

When you register a plug-in, you can store custom information in the plug-in configuration entry. When the server invokes the plug-in, it passes this information to the plug-in field, `PluginFlexfield`.

There are three schema attributes for storing custom information in the configuration entry. You can store text information in the `orclPluginFlexfield` attribute. You can use sub-types to provide more meaning to the kind of custom information being stored. For example, you could use the subtype `orclPluginFlexfield;ad-host` to store the host name of an Active Directory server that the plug-in must connect to.

You can store a binary value in the `orclPluginBinaryFlexfield` attribute. You can only store one value in `orclPluginBinaryFlexfield` for a plug-in because the server does not support attribute subtypes for binary attributes.

You can use `orclPluginSecuredFlexfield` to store custom text information that must never be displayed in clear text. The value is stored and displayed in encrypted form. Be sure that Oracle Internet Directory has privacy mode enabled to ensure that users cannot retrieve this attribute in clear text. See Enabling Privacy Mode of Sensitive Attributes. You can use sub-types to provide more meaning to the kind of custom information being stored. Use the same subtype format as for `orclPluginFlexfield`.

When the server invokes the plug-in, it passes the information from the `orclPluginFlexfield`, `orclPluginBinaryFlexfield`, and `orclPluginSecuredFlexfield` to the `PluginFlexfield` object. The plug-in can interpret the information and use it. It cannot return the `PluginFlexfield` to the server.

In the following configuration entry example, subtypes of `orclPluginFlexfield` specify that the minimum password length is 8 characters, that the password must contain a digit, and that the password cannot contain repeated characters:

```
dn: cn=pre_add replace,cn=plugin,cn=subconfigsubentry
orclPluginFlexfield;minPwdLength: 8
orclPluginFlexfield;isDigitPwd: 1
orclPluginFlexfield;isRepeatCharsPwd: 0
objectclass: orclPluginConfig
objectclass: top
orclpluginname: MyJavaPwdCheckPlugin
orclplugintype: configuration
orclplugintiming: pre
orclpluginldapoperation: ldapadd
orclpluginenable: 1
orclpluginsubscriberdnlist: cn=users,dc=us,dc=oracle,dc=com
orclpluginattributelist: userpassword
orclPluginKind: Java
```

A.4.4.4 PluginResult

To return the results of its execution to the server, a Java plug-in constructs a `PluginResult` object and passes it back to the server. The `PluginResult` contains one object: an `LdapOperation` or one of its operation-specific subclasses. These objects were described in LdapOperation Behavior As explained in that section, for some operations and timings, the plug-in can modify the information in the "LdapOperation Behavior" subclass object it received in the `PluginDetail` and send that object back to the server in the `PluginResult`.

A.4.4.5 ServerPlugin Interface Methods for LDAP Operations

All Java plug-ins use the `ServerPlugin` interface. The interface has pre-defined methods to communicate with the server. It has one method for each LDAP operation and timing. Each method takes a `PluginDetail` object as input and returns a `PluginResult` object back to the Oracle Internet Directory Server.

The `ServerPluginAdapter` class implements the `ServerPlugin` interface. The `ServerPluginAdapter` class has default (NULL) implementations of the `ServerPlugin` methods. This class enables you to code a Java plug-in without having to implement every method.

The rest of this section lists the `ServerPlugin` methods for each LDAP operation. It includes:

- ServerPlugin Methods for Ldapbind
- ServerPlugin Methods for Ldapcompare
- ServerPlugin Methods for Ldapadd
- ServerPlugin Methods for Ldapmodify
- ServerPlugin Methods for Ldapmoddn
- ServerPlugin Methods for Ldapsearch
- ServerPlugin Methods for Ldapdelete

A.4.4.5.1 ServerPlugin Methods for Ldapbind

```
public PluginResult pre_bind(PluginDetail pc) throws Exception;
public PluginResult when_bind_replace(PluginDetail pc) throws Exception;
public PluginResult post_bind(PluginDetail pc) throws Exception;
```

A.4.4.5.2 ServerPlugin Methods for Ldapcompare

```
public PluginResult pre_compare(PluginDetail pc) throws Exception;
public PluginResult when_compare_replace(PluginDetail pc) throws Exception;
public PluginResult post_compare(PluginDetail pc) throws Exception;
```

A.4.4.5.3 ServerPlugin Methods for Ldapadd

```
public PluginResult pre_add(PluginDetail pc) throws Exception;
public PluginResult when_add(PluginDetail pc) throws Exception;
public PluginResult when_add_replace(PluginDetail pc) throws Exception;
public PluginResult post_add(PluginDetail pc) throws Exception;
```

A.4.4.5.4 ServerPlugin Methods for Ldapmodify

```
public PluginResult pre_modify(PluginDetail pc) throws Exception;
public PluginResult when_modify(PluginDetail pc) throws Exception;
public PluginResult when_modify_replace(PluginDetail pc) throws Exception;
public PluginResult post_modify(PluginDetail pc) throws Exception;
```

A.4.4.5.5 ServerPlugin Methods for Ldapmoddn

```
public PluginResult pre_moddn(PluginDetail pc) throws Exception;
public PluginResult when_moddn(PluginDetail pc) throws Exception;
public PluginResult when_moddn_replace(PluginDetail pc) throws Exception;
public PluginResult post_moddn(PluginDetail pc) throws Exception;
```

A.4.4.5.6 ServerPlugin Methods for Ldapsearch

```
public PluginResult pre_search(PluginDetail pc) throws Exception;
public PluginResult when_search(PluginDetail pc) throws Exception;
public PluginResult when_search_replace(PluginDetail pc) throws Exception;
public PluginResult post_search(PluginDetail pc) throws Exception;
```

A.4.4.5.7 ServerPlugin Methods for Ldapdelete

```
public PluginResult pre_delete(PluginDetail pc) throws Exception;
public PluginResult when_delete(PluginDetail pc) throws Exception;
public PluginResult when_delete_replace(PluginDetail pc) throws Exception;
public PluginResult post_delete(PluginDetail pc) throws Exception;Java plug-in API
```

A.4.5 Java Plug-in Error and Exception Handling Examples

The Oracle Internet Directory server catches all unhandled exceptions during the execution of the plug-in. The exception stack trace and message for each exception is logged in the server log file.

These exceptions fall into three categories:

- Run-time errors and exceptions occur due to faulty plug-in code or logic. The server catches all run-time errors and exceptions, including `NullPointerException`s, raised during the execution of the Java plug-in. These errors and exceptions are logged in the server log file.
- Expected exceptions thrown by the plug-in are logged in the Oracle Internet Directory server log file. In addition, a plug-in can catch an exception and throw it back to the server to log it in the server log file.

- A plug-in can use the `PluginException` class to raise an error. The error message passed to the server with the `PluginException` object or its subclasses is passed on to the LDAP client. The server also logs this message in the server log file along with the exception stack trace and message.

This section includes three examples. They are:

- Run-time Exception Example
- Run-time Error Example
- PluginException Example

A.4.5.1 Run-time Exception Example

The log entry for a typical exception raised during execution of a plug-in looks like this:

```
….
06:17:03 *
    ERROR * gslpg_exceptionHndlr * Exception Message : Error
    ERROR * gslpg_exceptionHndlr * Exception Stack Trace :
              MyCompareJavaPlugin.post_compare(Prog2.java:75)
END

BEGIN
2004/10/19:01:52:13 *
    ServerWorker (REG):4 * ConnID:0 * OpID:1 * OpName:compare
    ERROR * gslpg_exceptionHndlr * Exception Stack Trace :
               java.lang.NullPointerException
              java.util.Hashtable.put(Hashtable.java:393)
               oracle.ldap.ospf.PluginDetail.put(PluginDetail.java:41)
END
```

A.4.5.2 Run-time Error Example

The error occurred because the plug-in `MyJavaPlugin` did not exist in the `$ORACLE_HOME/ldap/server/plugin` directory. The log file entry looks like this:

```
BEGIN
2004/10/19:01:52:13 *
    ServerWorker (REG):4 * ConnID:0 * OpID:1 * OpName:compare
    ERROR * gslpg_exceptionHndlr * Exception Stack Trace :
               java.lang.NoClassDefFoundError: MyJavaPlugin
END
```

A.4.5.3 PluginException Example

The Oracle Internet Directory server returns the standard plug-in error message to the LDAP client along with the additional error message if a `PluginException` object is thrown back to the server. The error displayed by the LDAP client looks something like this:

```
ldap_compare: UnKnown Error Encountered
ldap_compare: additional info: Error Message returned by the Java Plug-in
```

A.4.6 Java Plug-in Debugging and Logging

A plug-in can maintain its own log file and log to it in real time. In addition, a plug-in can log debug messages in the Oracle Internet Directory server log file during execution by using the `ServerLog` class.

The method for logging messages in the `ServerLog` class is:

```
public static void log(String message);
```

Messages logged by the `ServerLog.log()` method are preceded by the string:

```
* Server Java Plug-in *
```

For example:

```
2006/05/11:01:11:28 * ServerWorker (REG):7
  ConnID:241 * mesgID:2 * OpID:1 * OpName:bind
01:11:28 * Server Java Plug-in * MESSAGE FROM PLUGIN
01:11:28 * Server Java Plug-in * Bind DN :
 cn=ad_user,cn=oiddvusers,cn=oraclecontext,dc=us,dc=oracle,dc=com
```

To log plug-in debug messages to the server log, you must start the Oracle Internet Directory server using one of the following debug levels:

Table A-18 Debug Levels for Java Plug-in Logging

Oracle Internet Directory Server Debug Level	Debug Level Meaning
134217728	All Java plug-in debug messages and internal server messages related to the Java plug-in framework
268435456	All messages passed by a Java plug-in using the `ServerLog` object.
402653184	Both of the above

The `ServerLog.log()` method is thread safe. Execution of this method can degrade performance.

A.4.7 Java Plug-in Examples

Understand about Java Plug-ins with help of examples.

This sections includes the following examples:

- Example 1: Password Validation Plug-in
- Example 2: External Authentication Plug-in for Active Directory

> **Note:**
>
> Do not use System.exit() in a Java plug-in. Doing so might lead to unpredictable behavior by the Oracle directory server.

A.4.7.1 Example 1: Password Validation Plug-in

This example illustrates a Java plug-in that validates a `userPassword` before the ldapmodify operation. A `pre` Java plug-in is registered with the Oracle Internet Directory server. The plug-in configuration includes the minimum password length to be checked for in the plug-in. This information is registered in the plug-in configuration entry using an `orclPluginFlexfield` attribute. The subtype `minPwdLength` specifies the minimum length. This information is passed to the plug-in using the `PluginFlexfield`. The `orclPluginName` specifies the name of the Java Plug-in to be invoked by the Oracle Internet Directory server.

The input to the plug-in is a `PluginDetail` and the output from the plug-in is a `PluginResult`.

This section contains the following topics:

- Password Validation Plug-in Configuration Entry
- Password Validation Plug-in Code Example

A.4.7.1.1 Password Validation Plug-in Configuration Entry

```
dn: cn=checkuserpassword,cn=plugin,cn=subconfigsubentry
orclPluginFlexfield;minPwdLength: 8
objectclass: orclPluginConfig
objectclass: top
orclpluginname: CheckPassword
orclplugintype: configuration
orclplugintiming: pre
orclpluginldapoperation: ldapmodify
orclpluginenable: 1
orclpluginsubscriberdnlist: cn=users,dc=us,dc=oracle,dc=com
orclpluginattributelist: userPassword
orclPluginKind: Java
```

A.4.7.1.2 Password Validation Plug-in Code Example

```
import java.io.*;
import java.lang.*;
import java.util.*;
import javax.naming.*;
import javax.naming.directory.*;
import oracle.ldap.ospf.*;
/**
 * This PRE modify plug-in will check whether the "userPassword"
 * is greater than 8 characters in length
 */
public class CheckPassword extends ServerPluginAdapter {

    // This PRE modify plug-in takes in a PluginDetail Object
    // and returns a PluginResult Object
    public PluginResult pre_modify(PluginDetail plgObj)
       throws Exception
    {
       try {
         // Retrieve the LdapOperation Object from the PluginDetail
         ModifyLdapOperation opObj = (ModifyLdapOperation)
plgObj.getLdapOperation();


         // Retrieve the LdapModification Object from the LdapOperation
            LdapModification modObj = opObj.getLdapModification();
```

```
         // Retrieve the PluginFlexfield Object from the PluginDetail
         PluginFlexfield flxFldObj = plgObj.getPluginFlexfield();

         // Retrieve the custom information from the PluginFlexfield
         // Get the minimum password length
 String passwdlength =
            flxFldObj.getFlexfield("minPwdLength");

        // Create a Result Object to return to the OID server
           PluginResult plgResObj = new PluginResult();

         // Check if the LdapModification Object is a NULL
         // set the appropriate error and error message
         if (modObj==null)
         {
            throw new PluginException("CheckPassword Plug-in Execution
Error");
         }

          // Retrieve the "userPassword" Attribute Value
         ModificationItem modItem = modObj.getModificationItemAt(0);
         BasicAttribute attr = (BasicAttribute)modItem.getAttribute();
         String attrval = null;
          if ((attr.getID()).equals("userpassword"))
     attrval = attr.get(0);

         // Check for the password length and set appropriate error
         // and error message
          if (attrval.length() < Integer.parseInt(passwdlength))
         {
             throw new PluginException("userPassword is less than 8
characters");
         }

        // Return the PluginResult Object to the OID Server
        return plgResObj;
    }
    // Catch any unexpected exception which may occur and throw
    // it back to the OID server to log it
    catch (Exception e)
    {
        throw e;
    }
  }
}
```

A.4.7.2 Example 2: External Authentication Plug-in for Active Directory

This example illustrates an external authentication plug-in for Active Directory. When a client requests an ldapcompare operation for `userPassword`, the server invokes this Java plug-in to authenticate the user against Active Directory.

This section contains the following topics:

- External Authentication Plug-in Configuration Entry
- External Authentication Plug-in Code

A.4.7.2.1 External Authentication Plug-in Configuration Entry

```
dn: cn=when_rep_comp,cn=plugin,cn=subconfigsubentry
orclpluginsubscriberdnlist: cn=users,dc=us,dc=oracle,dc=com;
orclpluginflexfield;ad-host: dlin-pc2.us.example.com
orclpluginflexfield;ad-port: 3060
orclpluginflexfield;ad-su-dn: administrator@dlin.net
orclpluginflexfield;ad-su-passwd: password1
objectclass: orclPluginConfig
objectclass: top
orclpluginname: ExtAuthAD
orclplugintype: configuration
orclplugintiming: when
orclpluginisreplace: 1
orclpluginldapoperation: ldapcompare
orclpluginversion: 1.0.1
cn: when_rep_comp
orclpluginkind: Java
orclpluginenable: 1
```

A.4.7.2.2 External Authentication Plug-in Code

```
public class ExtAuthAD extends ServerPluginAdapter {

public PluginResult when_compare_replace(PluginDetail plgObj)
   throws Exception {
   try {

       // Retrieve the LdapOperation from the PluginDetail
       CompareLdapOperation opObj =
           (CompareLdapOperation) plgObj.getLdapOperation();       String baseDn =
plgObj.getLdapBaseEntry().getDN();

       // Retrieve the Base DN, Attribute and Attribute Value
       String bdn = baseDn.substring(0,
             baseDn.lastIndexOf("cn=users,dc=us,dc=oracle,dc=com")-1)
             +",cn=users,dc=dlin,dc=net";
       String ban = opObj.getAttributeName();       String bav =
opObj.getAttributeValue().toString();

       // Retrieve the AD Information from the PluginFlexfield
       PluginFlexfield flxObj = plgObj.getPluginFlexfield();
       String adhost = flxObj.getFlexfield("ad-host");
       String adport = flxObj.getFlexfield("ad-port");
       String adsudn = flxObj.getFlexfield("ad-su-dn");
       String adsupasswd = flxObj.getFlexfield("ad-su-passwd");

       // Create a PluginResult Object to return to the OID server
       PluginResult plgResObj = new PluginResult();


       // Create a Hashtable with values required to connect to AD
       Hashtable env = new Hashtable();

       env.put(Context.INITIAL_CONTEXT_FACTORY,
               "com.sun.jndi.ldap.LdapCtxFactory");
       env.put(Context.PROVIDER_URL, "ldap://"+adhost+":"+adport);
       env.put(Context.SECURITY_AUTHENTICATION, "simple");
       env.put(Context.SECURITY_PRINCIPAL, bdn);
       env.put(Context.SECURITY_CREDENTIALS, bav);
```

```
        // Try to connect to AD
        DirContext dirContext = null;
        try {
          dirContext = new InitialDirContext(env);
          if (dirContext != null) {
              // User has been successfully authenticated, add the appropriate
              // result code to the LdapOperation
              opObj.setOperationResultCode(6);
          }
        }
        catch(NamingException ne) {
         // Unable to connect to the AD directory server with the given
         // credentials, add the appropriate result code to the LdapOperation
          opObj.setOperationResultCode(5);
        }

        // Add the LdapOperation to the PluginResult
        plgResObj.setLdapOperation(opObj);

        // Return the PluginResult
        return plgResObj;

    } catch(Exception e) {
       // In case of any unexpected errors in the plug-in, throw the Exception
       // back to the OID server to log it     throw e;
     }
    }
  }
```

A.5 PL/SQL Server Plug-in Developer's Reference

You can use the plug-in framework in PL/SQL, including designing, creating, and using PL/SQL server plug-ins with Oracle Internet Directory.
The following sections explain this further:

- Designing, Creating, and Using PL/SQL Server Plug-ins
- Using PL/SQL Plug-ins
- Performing Binary Operations by using PL/SQLPlug-ins
- Object Type Definitions in the LDAP API Plug-in
- Specifications for PL/SQL Plug-in Procedures

A.5.1 Designing, Creating, and Using PL/SQL Server Plug-ins

You can design, create, and use PL/SQL Server plug-ins.

This section contains the following topics:

- PL/SQLPlug-in Caveats
- Specifications for PL/SQL Plug-in Package Names and Procedures
- Compiling PL/SQLPlug-ins
- Managing PL/SQL Plug-ins

- Enabling and Disabling PL/SQL Plug-ins
- Exception Handling in a PL/SQL Plug-in
- PL/SQL Plug-in LDAP API
- PL/SQL Plug-in and Database Tools
- Ensuring Security in PL/SQL Plug-ins
- PL/SQL Plug-in Debugging
- Specifications for PL/SQL Plug-in LDAP API

A.5.1.1 PL/SQLPlug-in Caveats

PL/SQL plug-ins can only be associated with specific operations. Also the plug-in names (PL/SQL package names) must be unique if they share the same database schema with other plug-ins or stored procedures. The following sections explain this further:

- Types of PL/SQL Plug-in Operations
- Unique PL/SQL Plug-in Names

A.5.1.1.1 Types of PL/SQL Plug-in Operations

A PL/SQL plug-in can only be associated with `ldapbind`, `ldapadd`, `ldapmodify`, `ldapcompare`, `ldapsearch`, and `ldapdelete` operations. You cannot associate a PL/SQL plug-in with `moddn`. If you must associate a plug-in with `moddn`, you must use a Java plug-in.

A.5.1.1.2 Unique PL/SQL Plug-in Names

Plug-in names (PL/SQL package names) must be unique if they share the same database schema with other plug-ins or stored procedures. But plug-ins can share names with other database schema objects such as tables and views. This kind of sharing is not, however, recommended.

A.5.1.2 Specifications for PL/SQL Plug-in Package Names and Procedures

Creating a PL/SQL plug-in module is like creating a PL/SQL package. Both have a specification part and a body part. The directory, not the plug-in, defines the plug-in specification because the specification serves as the interface between Oracle Internet Directory and the custom plug-in.

For security reasons and for the integrity of the LDAP server, you can compile PL/SQL plug-ins only in the ODS database schema. You must compile them in the database that serves as the back end database of Oracle Internet Directory. The following sections explain it further:

- Specifications for Plug-in Package Names
- Specifications for Plug-in Procedures

A.5.1.2.1 Specifications for Plug-in Package Names

Different plug-ins have different package specifications. As Table A-19 shows, you can name the plug-in package. You must, however, follow the signatures defined for each type of plug-in procedure. See Specifications for PL/SQL Plug-in Procedures for details.

Table A-19 Plug-in Module Interface

Plug-in Item	User Defined	Oracle Internet Directory-Defined
Plug-in Package Name	X	
Plug-in Procedure Name		X
Plug-in Procedure Signature		X

A.5.1.2.2 Specifications for Plug-in Procedures

Table A-20 names the different plug-in procedures. In addition, it lists and describes the parameters that these procedures use.

Table A-20 Operation-Based and Attribute-Based Plug-in Procedure Signatures

Invocation Context	Procedure Name	IN Parameters	OUT Parameters
Before `ldapbind`	`PRE_BIND`	ldapcontext, Bind DN, Password	return code, error message
With `ldapbind` but replacing the default server behavior	`WHEN_BIND_REPLACE`	ldapcontext, bind result, DN, userpassword	bind result, return code, error message
After `ldapbind`	`POST_BIND`	ldapcontext, Bind result, Bind DN, Password	return code, error message
Before `ldapmodify`	`PRE_MODIFY`	ldapcontext, DN, Mod structure	return code, error message
With `ldapmodify`	`WHEN_MODIFY`	ldapcontext, DN, Mod structure	return code, error message
With `ldapmodify` but replacing the default server behavior	`WHEN_MODIFY_REPLACE`	ldapcontext, DN, Mod structure	return code, error message
After `ldapmodify`	`POST_MODIFY`	ldapcontext, Modify result, DN, Mod structure	return code, error message
Before `ldapcompare`	`PRE_COMPARE`	ldapcontext, DN, attribute, value	return code, error message
With `ldapcompare` but replacing the default server behavior	`WHEN_COMPARE_REPLACE`	ldapcontext, Compare result, DN, attribute, value	compare result, return code, error message
After `ldapcompare`	`POST_COMPARE`	ldapcontext, Compare result, DN, attribute, value	return code, error message
Before `ldapadd`	`PRE_ADD`	ldapcontext, DN, Entry	return code, error message
With `ldapadd`	`WHEN_ADD`	ldapcontext, DN, Entry	return code, error message
With `ldapadd` but replacing the default server behavior	`WHEN_ADD_REPLACE`	ldapcontext, DN, Entry	return code, error message

Table A-20 (Cont.) Operation-Based and Attribute-Based Plug-in Procedure Signatures

Invocation Context	Procedure Name	IN Parameters	OUT Parameters
After `ldapadd`	`POST_ADD`	ldapcontext, Add result, DN, Entry	return code, error message
Before `ldapdelete`	`PRE_DELETE`	ldapcontext, DN	return code, error message
With `ldapdelete`	`WHEN_DELETE`	ldapcontext, DN	return code, error message
With `ldapdelete` but replacing the default server behavior	`WHEN_DELETE`	ldapcontext, DN	return code, error message
After `ldapdelete`	`POST_DELETE`	ldapcontext, Delete result, DN	return code, error message
Before `ldapsearch`	`PRE_SEARCH`	ldapcontext, Base DN, scope, filter	return code, error message
After `ldapsearch`	`POST_SEARCH`	Ldap context, Search result, Base DN, scope, filter	return code, error message

See Also:

- Error Messages and Return Codes in OID for valid values for the return code and error message.
- Specifications for PL/SQL Plug-in Procedures for complete supported procedure signatures.

A.5.1.3 Compiling PL/SQLPlug-ins

You must compile the plug-in module against the same database that serves as the Oracle Internet Directory back end database. Plug-ins are the same as PL/SQL stored procedures. A PL/SQL anonymous block is compiled each time it is loaded into memory.

All plug-in modules must be compiled in the ODS database schema. The following sections explain it further:

- Overview of Stages in Plug-in Compilation
- Viewing Errors Occurred During Compilation of Plug-ins
- Compiled Plug-in Dependencies
- Recompiling Plug-ins

A.5.1.3.1 Overview of Stages in Plug-in Compilation

Compilation consists of three stages. These are:

- Syntax checking: PL/SQL syntax is checked, and a parse tree is generated.
- Semantic checking: Type checking and further processing on the parse tree.

- Code generation: The pcode is generated.

A.5.1.3.2 Viewing Errors Occurred During Compilation of Plug-ins

If errors occur during the compilation of a plug-in, the plug-in is not created. You can use the `SHOW ERRORS` statement in SQL*Plus or Enterprise Manager to see any compilation errors when you create a plug-in, or you can `SELECT` the errors from the `USER_ERRORS` view.

A.5.1.3.3 Compiled Plug-in Dependencies

Compiled plug-ins have dependencies. They become invalid if an object depended upon, such as a stored procedure or function called from the plug-in body, is modified. Plug-ins that are invalidated for dependency reasons must be recompiled before the next invocation.

A.5.1.3.4 Recompiling Plug-ins

Use the `ALTER PACKAGE` statement to manually recompile a plug-in. For example, the following statement recompiles the my_plugin plug-in:

```
ALTER PACKAGE my_plugin COMPILE PACKAGE;
```

A.5.1.4 Managing PL/SQL Plug-ins

The following topics explain how to modify and debug plug-ins.

- Modifying Plug-ins
- Debugging Plug-ins

A.5.1.4.1 Modifying Plug-ins

Like a stored procedure, a plug-in cannot be explicitly altered. It must be replaced with a new definition.

When replacing a plug-in, you must include the `OR REPLACE` option in the `CREATE PACKAGE` statement. The `OR REPLACE` option enables a new version of an existing plug-in to replace an older version without having an effect on grants made for the original version of the plug-in.

Alternatively, the plug-in can be dropped using the `DROP PACKAGE` statement, and you can rerun the `CREATE PACKAGE` statement.

If the plug-in name (the package name) is changed, you must register the new plug-in again.

A.5.1.4.2 Debugging Plug-ins

You can debug a plug-in using the same facilities available for PL/SQL stored procedures.

A.5.1.5 Enabling and Disabling PL/SQL Plug-ins

To turn the plug-in on or off, modify the value of `orclPluginEnable` in the plug-in configuration object. For example, modify the value of `orclPluginEnable` in `cn=post_mod_plugin,cn=plugins,cn=subconfigsubentry` to be `1` or `0`.

A.5.1.6 Exception Handling in a PL/SQL Plug-in

Each of the procedures in a PL/SQL plug-in must have an exception handling block that handles errors intelligently and, if possible, recovers from them. The following sections explain it further:

- Error Messages and Return Codes in OID
- Action in Oracle Internet Directory Server in Response to Plug-in Exception or Failure

A.5.1.6.1 Error Messages and Return Codes in OID

Oracle Internet Directory requires that the return code (`rc`) and error message (`errmsg`) be set correctly in the plug-in procedures.

- Return Codes for Error Handling
- Displaying and Logging Error Messages

A.5.1.6.1.1 Return Codes for Error Handling

Table A-21 provides the values that are valid for the return code.

Table A-21 Valid Values for the plug-in Return Code

Error Code	Description
`0`	Success
Any number greater than zero	Failure
`-1`	Warning

A.5.1.6.1.2 Displaying and Logging Error Messages

The `errmsg` parameter is a string value that can pass a user's custom error message back to Oracle Internet Directory server. The size limit for `errmsg` is 1024 bytes. Each time Oracle Internet Directory runs the plug-in program, it examines the return code to determine if it must display the error message.

If, for example, the value for the return code is `0`, the error message value is ignored. If the value of the return code is `-1` or greater than zero, the following message is either logged in the log file or displayed in standard output if the request came from LDAP command-line tools:

```
ldap addition info: customized error
```

A.5.1.6.2 Action in Oracle Internet Directory Server in Response to Plug-in Exception or Failure

Oracle Internet Directory Server handles exceptions in different ways. The following sections explain it further:

- Action in Oracle Internet Directory Server when a Plug-in Exception Occurs
- Action in Oracle Internet Directory Server when a LDAP Operation Fails

A.5.1.6.2.1 Action in Oracle Internet Directory Server when a Plug-in Exception Occurs

Table A-22 shows where plug-in exceptions occur and how the directory handles them.

Table A-22 Program Control Handling when a Plug-in Exception Occurs

Plug-in Exception Occurred in	Oracle Internet Directory Server Handling
`PRE_BIND`, `PRE_MODIFY`, `PRE_ADD`, `PRE_SEARCH`, `PRE_COMPARE`, `PRE_DELETE`	Depends on return code. If the return code is: • Greater than zero (error), then no LDAP operation is performed • `-1` (warning), then proceed with the LDAP operation
`POST_BIND`, `POST_MODIFY`, `POST_ADD`, `POST_SEARCH`, `WHEN_DELETE`	LDAP operation is completed. There is no rollback.
`WHEN_MODIFY`, `WHEN_ADD`, `WHEN_DELETE`	Rollback the LDAP operation

A.5.1.6.2.2 Action in Oracle Internet Directory Server when a LDAP Operation Fails

Table A-23 shows how the directory responds when an LDAP operation fails.

Table A-23 Program Control Handling when an LDAP Operation Fails

LDAP Operation Fails in	Oracle Internet Directory Server Handling
`PRE_BIND`, `PRE_MODIFY`, `PRE_ADD`, `PRE_SEARCH`, `WHEN_DELETE`	Pre-operation plug-in is completed. There is no rollback.
`POST_BIND`, `POST_MODIFY`, `POST_ADD`, `POST_SEARCH`, `WHEN_DELETE`	Proceed with post-operation plug-in. The LDAP operation result is one of the `IN` parameters.
`WHEN_MODIFY`, `WHEN_ADD`, `WHEN_DELETE`	When types of plug-in changes are rolled back.
`WHEN`	Changes made in the plug-in program body are rolled back.

A.5.1.7 PL/SQL Plug-in LDAP API

There are different methods for providing API access:

- Enable a user to utilize the standard LDAP PL/SQL APIs. Note though that, if program logic is not carefully planned, an infinite loop in plug-in execution can result.
- Oracle Internet Directory provides the Plug-in LDAP API. This plug-in does not cause a series of plug-in actions in the directory server if there are plug-ins configured and associated with the LDAP request.

In the Plug-in LDAP API, the directory provides APIs for connecting back to the directory server designated in the plug-in module. You must use this API if you want to connect to the server that is executing the plug-in. If you want to connect to an external server, you can use the DBMS_LDAP API.

Within each plug-in module, an `ldapcontext` is passed from the Oracle directory server. When the Plug-in LDAP API is called, `ldapcontext` is passed for security and binding purposes.

When binding with this `ldapcontext`, Oracle Internet Directory recognizes that the LDAP request is coming from a plug-in module. For this type of plug-in bind, the directory does not trigger any subsequent plug-ins. It handles the plug-in bind as a super-user bind. Use this plug-in bind with discretion.

> **See Also:**
>
> Specifications for PL/SQL Plug-in LDAP API .

A.5.1.8 PL/SQL Plug-in and Database Tools

Bulk tools do not support server plug-ins.

A.5.1.9 Ensuring Security in PL/SQL Plug-ins

Some Oracle Internet Directory server plug-ins require that you supply the code that preserves tight security. For example, if you replace the directory's `ldapcompare` or `ldapbind` operation with your own plug-in module, you must ensure that your implementation of this operation does not omit any functionality on which security relies.

To ensure tight security, the following must be done:

- Create the plug-in packages
- Only the LDAP administrator can restrict the database user
- Use the access control list (ACL) to set the plug-in configuration entries to be accessed only by the LDAP administrator
- Be aware of the program relationship between different plug-ins

A.5.1.10 PL/SQL Plug-in Debugging

Use the plug-in debugging mechanism for Oracle Internet Directory to examine the process and content of plug-ins.The following commands control the operation of the server debugging process:

- To set up plug-in debugging, run this command:

  ```
  % sqlplus ods @$ORACLE/ldap/admin/oidspdsu.pls
  ```

- To enable plug-in debugging, run this command:

  ```
  % sqlplus ods @$ORACLE/ldap/admin/oidspdon.pls
  ```

- After enabling plug-in debugging, you can use this command in the plug-in module code:

  ```
  plg_debug('debuggingmessage');
  ```

  The resulting debug message is stored in the plug-in debugging table.

- To disable debugging, run this command:

  ```
  % sqlplus ods @$ORACLE/ldap/admin/oidspdof.pls
  ```

- To display the debug messages that you put in the plug-in module, run this command:

  ```
  % sqlplus ods @$ORACLE/ldap/admin/oidspdsh.pls
  ```

- To delete all of the debug messages from the debug table, run this command:

```
% sqlplus ods @$ORACLE/ldap/admin/oidspdde.pls
```

A.5.1.11 Specifications for PL/SQL Plug-in LDAP API

Here is the package specification that Oracle Internet Directory provides for the PL/SQL Plug-in LDAP API:

```
CREATE OR REPLACE  PACKAGE LDAP_PLUGIN AS
    SUBTYPE SESSION IS RAW(32);

    -- Initializes the LDAP library and return a session handler
    -- for use in subsequent calls.
    FUNCTION init (ldappluginctx IN ODS.plugincontext)
      RETURN SESSION;

    -- Synchronously authenticates to the directory server using
    -- a Distinguished Name and password.
    FUNCTION simple_bind_s (ldappluginctx IN ODS.plugincontext,
                            ld            IN SESSION)
      RETURN PLS_INTEGER;

    -- Get requester info from the plug-in context
    FUNCTION get_requester (ldappluginctx IN ODS.plugincontext)
      RETURN VARCHAR2;
END LDAP_PLUGIN;
```

A.5.2 Using PL/SQL Plug-ins

Understand about PL/SQL Plug-ins using examples. This section presents two sample plug-ins. One logs all `ldapsearch` commands. The other synchronizes two directory information trees (DITs).

This section contains the following topics:

- Logging all ldapsearch Commands through a Plug-in
- Synchronizing Two DITs through a Plug-in

A.5.2.1 Logging all ldapsearch Commands through a Plug-in

Situation: A user wants to know if it is possible to log all of the `ldapsearch` commands.

Solution: Yes. The user can use the post `ldapsearch` configuration plug-in for this purpose. They can either log all of the requests or only those that occur under the DNs being searched.

To log all the `ldapsearch` commands:

1. Log all of the `ldapsearch` results into a database table. This log table has these columns:
   - timestamp
   - baseDN
   - search scope
   - search filter
   - required attribute
   - search result

   Use this SQL script to create the table:

```
drop table search_log;
create table search_log
(timestamp varchar2(50),
basedn varchar2(256),
searchscope number(1);
searchfilter varchar2(256);
searchresult number(1));
drop table simple_tab;
create table simple_tab (id NUMBER(7), dump varchar2(256));
DROP sequence seq;
CREATE sequence seq START WITH 10000;
commit;
```

2. Create the plug-in package specification.

```
CREATE OR REPLACE PACKAGE LDAP_PLUGIN_EXAMPLE1 AS
PROCEDURE post_search
(ldapplugincontext IN  ODS.plugincontext,
result       IN  INTEGER,
baseDN       IN  VARCHAR2,
scope        IN  INTEGER,
filterStr    IN  VARCHAR2,
requiredAttr IN  ODS.strCollection,
rc           OUT INTEGER,
errormsg     OUT VARCHAR2
);
END LDAP_PLUGIN_EXAMPLE1;
/
```

3. Create the plug-in package body.

```
CREATE OR REPLACE PACKAGE BODY LDAP_PLUGIN_EXAMPLE1 AS
PROCEDURE post_search
(ldapplugincontext IN  ODS.plugincontext,
result       IN  INTEGER,
baseDN       IN  VARCHAR2,
scope        IN  INTEGER,
filterStr    IN  VARCHAR2,
requiredAttr IN  ODS.strCollection,
rc           OUT INTEGER,
errormsg     OUT VARCHAR2
)
      IS
BEGIN
   INSERT INTO simple_tab VALUES
(to_char(sysdate, 'Month DD, YYYY HH24:MI:SS'), baseDN, scope, filterStr, result);
   -- The following code segment demonstrate how to iterate
   -- the ODS.strCollection
   FOR l_counter1 IN 1..requiredAttr.COUNT LOOP
      INSERT INTO simple_tab
        values (seq.NEXTVAL, 'req attr ' || l_counter1 || ' = ' ||
                requiredAttr(l_counter1));
   END LOOP;
   rc := 0;
   errormsg := 'no post_search plug-in error msg';
   COMMIT;
EXCEPTION
   WHEN others THEN
      rc := 1;
      errormsg := 'exception: post_search plug-in';
END;
END LDAP_PLUGIN_EXAMPLE1;
/
```

4. Register the plug-in entry in Oracle Internet Directory.

```
dn: cn=post_search,cn=plugin,cn=subconfigsubentry
objectclass: orclPluginConfig
objectclass: top
orclPluginName: ldap_plugin_example1
orclPluginType: configuration
orclPluginTiming: post
orclPluginLDAPOperation: ldapsearch
orclPluginEnable: 1
orclPluginVersion: 1.0.1
cn: post_search
orclPluginKind: PLSQL
```

Using the `ldapadd` command-line tool to add this entry:

```
% ldapadd -p port_number -h host_name -D bind_dn -q -v \
          -f register_post_search.ldif
```

A.5.2.2 Synchronizing Two DITs through a Plug-in

Situation: There are two interdependent products under `cn=Products`, `cn=oraclecontext`. This interdependency extends down to the users in these products' containers. If a user in the first DIT (product 1) is deleted, the corresponding user in the other DIT (product 2) must be deleted.

Is it possible to set a trigger that, when the user in the first DIT is deleted, calls or passes a trigger to delete the user in the second DIT?

Solution: Yes, we can use the post `ldapdelete` operation plug-in to handle the second deletion occurring in the second DIT.

If the first DIT has the naming context of `cn=DIT1,cn=products,cn=oraclecontext` and the second DIT has the naming context of `cn=DIT2,cn=products,cn=oraclecontext`, the two users share the same ID attribute. Inside of the post `ldapdelete` plug-in module, we can use `LDAP_PLUGIN` and `DBMS_LDAP` APIs to delete the user in the second DIT.

We must set `orclPluginSubscriberDNList` to `cn=DIT1,cn=products,cn=oraclecontext`, so that whenever we delete entries under `cn=DIT1,cn=products,cn=oraclecontext`, the plug-in module is invoked.

> **Note:**
>
> When you use a post `ldapmodify` plug-in to synchronize changes between two Oracle Internet Directory nodes, you cannot push all the attributes from one node to the other. This is because the changes (mod structure) captured in the plug-in module include operational attributes. These operational attributes are generated on each node and cannot be modified by using the standard LDAP methods.
>
> When writing your plug-in program, exclude the following operational attributes from synchronization: `authPassword`, `creatorsname`, `createtimestamp`, `modifiersname`, `modifytimestamp`, `orcllastlogintime`, `pwdchangedtime`, `pwdfailuretime`, `pwdaccountlockedtime`, `pwdexpirationwarned`, `pwdreset`, `pwdhistory`, `pwdgraceusetime`.

To synchronize:

1. Assume that the entries under both DITs have been added to the directory. For example, the entry `id=12345,cn=DIT1,cn=products,cn=oraclecontext` is in `DIT1`, and `id=12345,cn=DIT2,cn=products,cn=oraclecontext` is in `DIT2`.
2. Create the plug-in package specification.

```
CREATE OR REPLACE PACKAGE LDAP_PLUGIN_EXAMPLE2 AS
PROCEDURE post_delete
(ldapplugincontext IN  ODS.plugincontext,
result   IN  INTEGER,
dn       IN  VARCHAR2,
rc       OUT INTEGER,
errormsg OUT VARCHAR2
);
END LDAP_PLUGIN_EXAMPLE2;
/
```

3. Create the plug-in package body.

```
CREATE OR REPLACE PACKAGE BODY LDAP_PLUGIN_EXAMPLE2 AS
PROCEDURE post_delete
(ldapplugincontext IN  ODS.plugincontext,
result   IN  INTEGER,
dn       IN  VARCHAR2,
rc       OUT INTEGER,
errormsg OUT VARCHAR2
)
    IS
     retval       PLS_INTEGER;
     my_session   DBMS_LDAP.session;
     newDN        VARCHAR2(256);
BEGIN
   retval         := -1;
   my_session := LDAP_PLUGIN.init(ldapplugincontext);
   -- bind to the directory
   retval := LDAP_PLUGIN.simple_bind_s(ldapplugincontext, my_session);
   -- if retval is not 0, then raise exception
   newDN := REPLACE(dn,'DIT1','DIT2');
   retval := DBMS_LDAP.delete_s(my_session, newDN);
   -- if retval is not 0, then raise exception
   rc := 0;
   errormsg := 'no post_delete plug-in error msg';
EXCEPTION
   WHEN others THEN
      rc := 1;
      errormsg := 'exception: post_delete plug-in';
END;
END LDAP_PLUGIN_EXAMPLE2;
/
(ldapplugincontext IN  ODS.plugincontext,
result   IN  INTEGER,
dn       IN  VARCHAR2,
rc       OUT INTEGER,
errormsg OUT VARCHAR2
)
    IS
     retval       PLS_INTEGER;
     my_session   DBMS_LDAP.session;
     newDN        VARCHAR2(256);
BEGIN
   retval         := -1;
   my_session := LDAP_PLUGIN.init(ldapplugincontext);
   -- bind to the directory
```

```
   retval := LDAP_PLUGIN.simple_bind_s(ldapplugincontext, my_session);
   -- if retval is not 0, then raise exception
   newDN := REPLACE(dn,'DIT1','DIT2');
   retval := DBMS_LDAP.delete_s(my_session, newDN);
   -- if retval is not 0, then raise exception
   rc := 0;
   errormsg := 'no post_delete plug-in error msg';
EXCEPTION
   WHEN others THEN
      rc := 1;
      errormsg := 'exception: post_delete plug-in';
END;
END LDAP_PLUGIN_EXAMPLE2;
/
```

4. Register the plug-in entry with Oracle Internet Directory.

   Construct the LDIF file `register_post_delete.ldif`:

```
dn: cn=post_delete,cn=plugin,cn=subconfigsubentry
objectclass: orclPluginConfig
objectclass: top
orclPluginName: ldap_plugin_example2
orclPluginType: configuration
orclPluginTiming: post
orclPluginLDAPOperation: ldapdelete
orclPluginEnable: 1
orclPluginSubscriberDNList: cn=DIT1,cn=oraclecontext,cn=products
orclPluginVersion: 1.0.1
cn: post_delete
orclPluginKind: PLSQL
```

   Use the `ldapadd` command-line tool to add this entry:

```
% ldapadd -p port_number -h host_name -D bind_dn -q -v -f register_
post_delete.ldif
```

A.5.3 Performing Binary Operations by using PL/SQLPlug-ins

Object definitions in the Plug-in LDAP API enable `ldapmodify`, `ldapadd`, and `ldapcompare` plug-ins to access binary attributes in the directory database.

Formerly, only attributes of type `VARCHAR2` could be accessed. These object definitions do not invalidate plug-in code that precedes release 10.1.2. No change to this code is required. The new definitions appear in Object Type Definitions in the LDAP API Plug-in.

The section that you are reading now examines binary operations involving the three types of plug-ins. It includes examples of these plug-ins. The new object definitions apply to pre, post, and when versions of all three.

Note that the three examples use RAW functions and variables in place of LOBs. The following sections explain this further:

- Modifying an Entry in Another Directory by using ldapmodify Plug-in
- Propagating a Change to Another Directory by using ldapadd Plug-in
- Comparing Binary Attributes by using ldapcompare Plug-in

A.5.3.1 Modifying an Entry in Another Directory by using ldapmodify Plug-in

The `modobj` object that the plug-in framework passes to an `ldapmodify` plug-in now holds the values of binary attributes as `binvals`. This variable is a table of `binvalobj` objects.

The plug-in determines whether a binary operation is being performed by examining the `operation` field of `modobj`. It checks whether any of the values `DBMS_LDAP.MOD_ADD`, `DBMS_LDAP.MOD_DELETE`, and `DBMS_LDAP.MOD_REPLACE` are paired with `DBMS_LDAP.MOD_BVALUES`. The pairing `DBMS_LDAP.MOD_ADD+DBMS_LDAP.MOD_BVALUES`, for example, signifies a binary add in the modify operation.

The example that follows shows a post `ldapmodify` plug-in modifying an entry in another directory. The plug-in is invoked after `ldapmodify` applies the same change to the same entry in the plug-in directory. The entry in the other directory appears under the DIT `cn=users,dc=us,dc=example,dc=com`.

```
create or replace package moduser as
   procedure post_modify(ldapplugincontext IN ODS.plugincontext,
                         result IN integer,
                         dn IN varchar2,
                         mods IN ODS.modlist,
                         rc OUT integer,
                         errormsg OUT varchar2);
end moduser;
/
show error

CREATE OR REPLACE PACKAGE BODY moduser AS
   procedure post_modify(ldapplugincontext IN ODS.plugincontext,
                         result IN integer,
                         dn IN varchar2,
                         mods IN ODS.modlist,
                         rc OUT integer,
                         errormsg OUT varchar2)
   is
      counter1 pls_integer;
      counter2 pls_integer;
      retval pls_integer := -1;
      user_session DBMS_LDAP.session;
      user_dn varchar(256);
      user_array DBMS_LDAP.mod_array;
      user_vals DBMS_LDAP.string_collection;
      user_binvals DBMS_LDAP.blob_collection;
      ldap_host varchar(256);
      ldap_port varchar(256);
      ldap_user varchar(256);
      ldap_passwd varchar(256);
   begin
      ldap_host :='backup.us.example.com';
      ldap_port :='4000';
      ldap_user :='cn=orcladmin';
      ldap_passwd :='password';

      plg_debug('START MODIFYING THE ENTRY');

      -- Get a session
         user_session := dbms_ldap.init(ldap_host, ldap_port);

      -- Bind to the directory
         retval := dbms_ldap.simple_bind_s(user_session, ldap_user,
```

```
   ldap_passwd);

-- Create a mod_array
   user_array := dbms_ldap.create_mod_array(mods.count);

-- Create a user_dn
   user_dn := substr(dn,1,instr(dn,',',1,1))||'cn=users,dc=us,dc=example,
   dc=com';

  plg_debug('THE CREATED DN IS'||user_dn);

-- Iterate through the modlist
   for counter1 in 1..mods.count loop

-- Log the attribute name and operation
   if (mods(counter1).operation > DBMS_LDAP.MOD_BVALUES) then
     plg_debug('THE NAME OF THE BINARY ATTR. IS'||mods(counter1).type);
   else
     plg_debug('THE NAME OF THE NORMAL ATTR. IS'||mods(counter1).type);
   end if;
     plg_debug('THE OPERATION IS'||mods(counter1).operation);

-- Add the attribute values to the collection
   for counter2 in 1..mods(counter1).vals.count loop
     user_vals(counter2) := mods(counter1).vals(counter2).val;
   end loop;

-- Add the attribute values to the collection
   for counter2 in 1..mods(counter1).binvals.count loop
     plg_debug('THE NO. OF BYTES OF THE BINARY ATTR. VALUE IS'
     ||mods(counter1).binvals(counter2).length);
     user_binvals(counter2) := mods(counter1).binvals(counter2).binval;
   end loop;

-- Populate the mod_array accordingly with binary/normal attributes
   if (mods(counter1).operation >= DBMS_LDAP.MOD_BVALUES) then
     dbms_ldap.populate_mod_array(user_array,mods(counter1).operation -
     DBMS_LDAP.MOD_BVALUES,mods(counter1).type,user_binvals);
     user_binvals.delete;
   else
     dbms_ldap.populate_mod_array(user_array,mods(counter1).operation,
     mods(counter1).type,user_vals);
     user_vals.delete;
   end if;

  end loop;

-- Modify the entry
   retval := dbms_ldap.modify_s(user_session,user_dn,user_array);
   if retval = 0 then
     rc := 0;
     errormsg:='No error occurred while modifying the entry';
   else
     rc := retval;
     errormsg :='Error code'||rc||' while modifying the entry';
   end if;

-- Free the mod_array
   dbms_ldap.free_mod_array(user_array);

 plg_debug('FINISHED MODIFYING THE ENTRY');
```

```
   exception
 WHEN others THEN
   plg_debug (SQLERRM);
 end;
end moduser;
/
show error

exit;
```

A.5.3.2 Propagating a Change to Another Directory by using ldapadd Plug-in

The `entryobj` object that the plug-in framework passes to an `ldapadd` plug-in now holds binary attributes as `binattr`. This variable is a table of `binattrobj` objects. The example that follows shows a post-add plug-in propagating a change (an added user) in the plug-in directory to another directory. In the latter directory, the entry appears under the DIT `cn=users,dc=us,dc=example,dc=com`.

```
create or replace package adduser as
   procedure post_add(ldapplugincontext IN ODS.plugincontext,
                          result IN integer,
                          dn IN varchar2,
                          entry IN ODS.entryobj,
                          rc OUT integer,
                          errormsg OUT varchar2);
end adduser;
/
show error

CREATE OR REPLACE PACKAGE BODY adduser AS
   procedure post_add(ldapplugincontext IN ODS.plugincontext,
                          result IN integer,
                          dn IN varchar2,
                          entry IN ODS.entryobj,
                          rc OUT integer,
                          errormsg OUT varchar2)
   is
      counter1 pls_integer;
      counter2 pls_integer;
      retval pls_integer := -1;
      s integer;
      user_session DBMS_LDAP.session;
      user_dn varchar(256);
      user_array DBMS_LDAP.mod_array;
      user_vals DBMS_LDAP.string_collection;
      user_binvals DBMS_LDAP.blob_collection;
      ldap_host varchar(256);
      ldap_port varchar(256);
      ldap_user varchar(256);
      ldap_passwd varchar(256);
   begin
      ldap_host :='backup.us.example.com';
      ldap_port :='4000';
      ldap_user :='cn=orcladmin';
      ldap_passwd :='password';

      plg_debug('START ADDING THE ENTRY');

      -- Get a session
         user_session := dbms_ldap.init(ldap_host, ldap_port);
```

```
        -- Bind to the directory
           retval := dbms_ldap.simple_bind_s(user_session, ldap_user, ldap_passwd);

        -- Create a mod_array
           user_array := dbms_ldap.create_mod_array(entry.binattr.count +
           entry.attr.count);

        -- Create a user_dn
           user_dn := substr(dn,1,instr(dn,',',1,1))||'cn=users,dc=us,dc=example,
           dc=com';
           plg_debug('THE CREATED DN IS'||user_dn);

        -- Populate the mod_array with binary attributes
           for counter1 in 1..entry.binattr.count loop
             for counter2 in 1..entry.binattr(counter1).binattrval.count loop
               plg_debug('THE NAME OF THE BINARY ATTR. IS'||
               entry.binattr(counter1).binattrname);
               s := dbms_lob.getlength(entry.binattr(counter1).
               binattrval(counter2));
               plg_debug('THE NO. OF BYTES OF THE BINARY ATTR. VALUE IS'||s);
               user_binvals(counter2) := entry.binattr(counter1).
               binattrval(counter2);
             end loop;
           dbms_ldap.populate_mod_array(user_array,DBMS_LDAP.MOD_ADD,
           entry.binattr(counter1).binattrname,user_binvals);
           user_binvals.delete;
         end loop;

        -- Populate the mod_array with attributes
           for counter1 in 1..entry.attr.count loop
             for counter2 in 1..entry.attr(counter1).attrval.count loop
               plg_debug('THE NORMAL ATTRIBUTE'||entry.attr(counter1).attrname||'
               HAS THE VALUE'||entry.attr(counter1).attrval(counter2));
               user_vals(counter2) := entry.attr(counter1).attrval(counter2);
             end loop;
           dbms_ldap.populate_mod_array(user_array,DBMS_LDAP.MOD_ADD,
           entry.attr(counter1).attrname,user_vals);
           user_vals.delete;
           end loop;

        -- Add the entry
           retval := dbms_ldap.add_s(user_session,user_dn,user_array);
           plg_debug('THE RETURN VALUE IS'||retval);
           if retval = 0 then
             rc := 0;
             errormsg:='No error occurred while adding the entry';
           else
             rc := retval;
             errormsg :='Error code'||rc||' while adding the entry';
           end if;

        -- Free the mod_array
           dbms_ldap.free_mod_array(user_array);
           retval := dbms_ldap.unbind_s(user_session);

           plg_debug('FINISHED ADDING THE ENTRY');

     exception
    WHEN others THEN
     plg_debug (SQLERRM);
    end;
   end adduser;
```

```
/
show error

exit;
```

A.5.3.3 Comparing Binary Attributes by using ldapcompare Plug-in

The `ldapcompare` plug-in can use three new overloaded module interfaces to compare binary attributes. If you want to use these interfaces to develop a plug-in package that handles both binary and nonbinary attributes, you must include two separate procedures in the package. The package name for both procedures is the same because only one `orclPluginName` can be registered in the plug-in entry.

After updating an existing plug-in package to include a procedure that compares binary attributes, reinstall the package. Recompile packages that depend on the plug-in package.

The three new interfaces look like this:

```
PROCEDURE pre_compare (ldapplugincontext IN ODS.plugincontext,
                       dn          IN VARCHAR2,
                       attrname    IN VARCHAR2,
                       attrval     IN BLOB,
                       rc          OUT INTEGER,
                       errormsg    OUT VARCHAR2 );

PROCEDURE when_compare_replace (ldapplugincontext IN ODS.plugincontext,
                                result      OUT INTEGER,
                                dn          IN VARCHAR2,
                                attrname    IN VARCHAR2,
                                attrval     IN BLOB,
                                rc          OUT INTEGER,
                                errormsg    OUT VARCHAR2 );

PROCEDURE post_compare (ldapplugincontext IN ODS.plugincontext,
                        result      IN INTEGER,
                        dn          IN VARCHAR2,
                        attrname    IN VARCHAR2,
                        attrval     IN BLOB,
                        rc          OUT INTEGER,
                        errormsg    OUT VARCHAR2 );
```

The example that follows compares a binary attribute of an entry in the plug-in directory with a binary attribute of an entry in another directory. This package replaces the compare code of the server with the compare code of the plug-in. The package handles both binary and nonbinary attributes. As such it contains two separate procedures.

```
create or replace package compareattr as
   procedure when_compare_replace(ldapplugincontext IN ODS.plugincontext,
                          result OUT integer,
                          dn IN varchar2,
                          attrname IN VARCHAR2,
                          attrval IN BLOB,
                          rc OUT integer,
                          errormsg OUT varchar2);
   procedure when_compare_replace(ldapplugincontext IN ODS.plugincontext,
                          result OUT integer,
                          dn IN varchar2,
                          attrname IN VARCHAR2,
                          attrval IN varchar2,
                          rc OUT integer,
                          errormsg OUT varchar2);
```

```
end compareattr;
/
show error

CREATE OR REPLACE PACKAGE BODY compareattr AS
   procedure when_compare_replace(ldapplugincontext IN ODS.plugincontext,
                          result OUT integer,
                          dn IN varchar2,
                          attrname IN VARCHAR2,
                          attrval IN varchar2,
                          rc OUT integer,
                          errormsg OUT varchar2)
   is
   pos            INTEGER := 2147483647;
   begin
      plg_debug('START');
      plg_debug('THE ATTRNAME IS'||attrname||' AND THE VALUE IS'||attrval);
      plg_debug('END');
      rc := 0;
      errormsg :='No error!!!';
   exception
   WHEN others THEN
    plg_debug ('Unknown UTL_FILE Error');
   end;

   procedure when_compare_replace(ldapplugincontext IN ODS.plugincontext,
                          result OUT integer,
                          dn IN varchar2,
                          attrname IN VARCHAR2,
                          attrval IN BLOB,
                          rc OUT integer,
                          errormsg OUT varchar2)
   is
      counter pls_integer;
      retval pls_integer := -1;
      cmp_result integer;
      s integer;
      user_session DBMS_LDAP.session;
      user_entry DBMS_LDAP.message;
      user_message DBMS_LDAP.message;
      user_dn varchar(256);
      user_attrs DBMS_LDAP.string_collection;
      user_attr_name VARCHAR2(256);
      user_ber_elmt DBMS_LDAP.ber_element;
      user_vals DBMS_LDAP.blob_collection;
      ldap_host varchar(256);
      ldap_port varchar(256);
      ldap_user varchar(256);
      ldap_passwd varchar(256);
      ldap_base varchar(256);
   begin
      ldap_host :='backup.us.example.com';
      ldap_port :='4000';
      ldap_user :='cn=orcladmin';
      ldap_passwd :='password';
      ldap_base := dn;

      plg_debug('STARTING COMPARISON IN WHEN REPLACE PLUG-IN');

        s := dbms_lob.getlength(attrval);
        plg_debug('THE NUMBER OF BYTES OF ATTRVAL'||s);
```

```
-- Get a session
   user_session := dbms_ldap.init(ldap_host, ldap_port);

-- Bind to the directory
   retval := dbms_ldap.simple_bind_s(user_session, ldap_user, ldap_passwd);

-- issue the search
   user_attrs(1) := attrname;
   retval := DBMS_LDAP.search_s(user_session, ldap_base,
                                DBMS_LDAP.SCOPE_BASE,
                               'objectclass=*',
                                user_attrs,
                                0,
                                user_message);

-- Get the entry in the other OID server
   user_entry := DBMS_LDAP.first_entry(user_session, user_message);

-- Log the DN and the Attribute name
   user_dn := DBMS_LDAP.get_dn(user_session, user_entry);
   plg_debug('THE DN IS'||user_dn);
   user_attr_name := DBMS_LDAP.first_attribute(user_session,user_entry,
   user_ber_elmt);

-- Get the values of the attribute
   user_vals := DBMS_LDAP.get_values_blob(user_session, user_entry,
   user_attr_name);

-- Start the binary comparison between the ATTRVAL and the attribute
-- values
   if user_vals.count > 0 then
    for counter in user_vals.first..user_vals.last loop
      cmp_result := dbms_lob.compare(user_vals(counter),attrval,
                      dbms_lob.getlength(user_vals(counter)),1,1);
      if  cmp_result = 0 then
           rc := 0;
           -- Return LDAP_COMPARE_TRUE
           result := 6;
           plg_debug('THE LENGTH OF THE ATTR.'||user_attr_name||' IN THE
           ENTRY IS'||dbms_lob.getlength(user_vals(counter)));
           errormsg :='NO ERROR. THE COMPARISON HAS SUCCEEDED.';
           plg_debug(errormsg);
           plg_debug('FINISHED COMPARISON');
           return;
      end if;
    end loop;
  end if;

rc := 1;
-- Return LDAP_COMPARE_FALSE
   result := 5;
   errormsg :='ERROR. THE COMPARISON HAS FAILED.';
   plg_debug('THE LENGTH OF THE ATTR.'||user_attr_name||' IN THE ENTRY IS'
   ||dbms_lob.getlength(user_vals(user_vals.last)));
   plg_debug(errormsg);
   plg_debug('FINISHED COMPARISON');

-- Free user_vals
   dbms_ldap.value_free_blob(user_vals);
   exception
   WHEN others THEN
     plg_debug (SQLERRM);
```

```
   end;
end compareattr;
/
show error

exit;
```

A.5.4 Object Type Definitions in the LDAP API Plug-in

Understand about the object types introduced in the Plug-in LDAP API. All of these definitions are in Oracle Directory Server database schema. Note that the API includes object types that enable plug-ins to extract binary data from the database.

```
create or replace type strCollection as TABLE of VARCHAR2(512);
/
create or replace type pluginContext as TABLE of VARCHAR2(512);
/
create or replace type attrvalType as TABLE OF VARCHAR2(4000);
/
create or replace type attrobj as object (
attrname    varchar2(2000),
attrval     attrvalType
);
/
create or replace type attrlist as table of attrobj;
/
create or replace type binattrvalType as TABLE OF BLOB;
/
create or replace type binattrobj as object (
binattrname    varchar2(2000),
binattrval     binattrvalType
);
/
create or replace type binattrlist as table of binattrobj;
/
create or replace type entryobj as object (
entryname      varchar2(2000),
attr           attrlist,
binattr        binattrlist
);
/
create or replace type entrylist as table of entryobj;
/

create or replace type bvalobj as object (
length         integer,
val            varchar2(4000)
);
/
create or replace type bvallist as table of bvalobj;
/
create or replace type binvalobj as object (
length         integer,
binval         blob
);
/
create or replace type binvallist as table of binvalobj;
/
create or replace type modobj as object (
operation      integer,
type           varchar2(256),
```

```
vals            bvallist,
binvals         binvallist
);
/
create or replace type modlist as table of modobj;
```

A.5.5 Specifications for PL/SQL Plug-in Procedures

When you use the plug-ins, you must adhere to the signature defined for each of them. Each signature is provided here.

```
PROCEDURE pre_add (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
entry           IN  ODS.entryobj,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);



PROCEDURE when_add (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
entry           IN  ODS.entryobj,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE when_add_replace (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
entry           IN  ODS.entryobj,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE post_add (ldapplugincontext IN  ODS.plugincontext,

result          IN  INTEGER,
dn              IN  VARCHAR2,
entry           IN  ODS.entryobj,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE pre_modify (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
mods            IN  ODS.modlist,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE when_modify (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
mods            IN  ODS.modlist,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE when_modify_replace (ldapplugincontext IN  ODS.plugincontext,
```

```
dn              IN  VARCHAR2,
mods            IN  ODS.modlist,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);


PROCEDURE post_modify (ldapplugincontext IN  ODS.plugincontext,

result          IN  INTEGER,
dn              IN  VARCHAR2,
mods            IN  ODS.modlist,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2);

PROCEDURE pre_compare (ldapplugincontext IN  ODS.plugincontext,

dn              IN  VARCHAR2,
attrname        IN  VARCHAR2,
attrval         IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE pre_compare (ldapplugincontext IN ODS.plugincontext,
dn              IN VARCHAR2,
attrname        IN VARCHAR2,
attrval         IN BLOB,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2 );

PROCEDURE when_compare_replace (ldapplugincontext IN  ODS.plugincontext,

result          OUT INTEGER,
dn              IN  VARCHAR2,
attrname        IN  VARCHAR2,
attrval         IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE when_compare_replace (ldapplugincontext IN ODS.plugincontext,
result          OUT INTEGER,
dn              IN VARCHAR2,
attrname        IN VARCHAR2,
attrval         IN BLOB,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2 );


PROCEDURE post_compare (ldapplugincontext IN  ODS.plugincontext,

result          IN  INTEGER,
dn              IN  VARCHAR2,
attrname        IN  VARCHAR2,
attrval         IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE post_compare (ldapplugincontext IN ODS.plugincontext,
result          IN INTEGER,
dn              IN VARCHAR2,
```

```
attrname        IN VARCHAR2,
attrval         IN BLOB,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2 );


PROCEDURE pre_delete (ldappluginContext IN  ODS.pluginContext,

dn              IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);


PROCEDURE when_delete (ldappluginContext IN  ODS.pluginContext,

dn              IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE when_delete_replace (ldappluginContext IN  ODS.pluginContext,

dn              IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);


PROCEDURE post_delete (ldappluginContext IN  ODS.pluginContext,

result          IN  INTEGER,
dn              IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);


PROCEDURE pre_search (ldappluginContext IN  ODS.pluginContext,

baseDN          IN  VARCHAR2,
scope           IN  INTEGER,
filterStr       IN  VARCHAR2,
requiredAttr    IN  ODS.strCollection,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE post_search (ldappluginContext IN  ODS.pluginContext,

result          IN  INTEGER,
baseDN          IN  VARCHAR2,
scope           IN  INTEGER,
filterStr       IN  VARCHAR2,
requiredAttr    IN  ODS.strCollection,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE pre_bind (ldappluginContext IN ODS.pluginContext,
```

```
dn              IN  VARCHAR2,
passwd          IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE when_bind_replace (ldapplugincontext IN ODS.plugincontext,

result          OUT INTEGER,
dn              IN  VARCHAR2,
passwd          IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);

PROCEDURE post_bind (ldapplugincontext IN ODS.plugincontext,

result          IN  INTEGER,
dn              IN  VARCHAR2,
passwd          IN  VARCHAR2,
rc              OUT INTEGER,
errormsg        OUT VARCHAR2
);
```

A.6 The LDAP Filter Definition

This appendix includes a paper titled "The String Representation of LDAP Search Filters." The paper contained in this appendix is copied with permission from RFC 2254 of the Internet Engineering Task Force. This document is located at: http://www.ietf.org

The contents of this paper may have been superseded by later papers or other information. Check the above Web site and related sites for additional or supplementary information.

> **Note:**
>
> ORACLE DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Network Working Group T. Howes Request for Comments: 2254 Netscape Communications Corp. Category: Standards Track December 1997

The String Representation of LDAP Search Filters

The following sections explain this further:

- Status of The String Representation of LDAP Search Filters
- IESG Note on The String Representation of LDAP Search Filters
- The String Representation of LDAP Search Filters Abstract
- LDAP Search Filter Definition
- String Search Filter Definition
- Using String Search Filters

- Security Considerations in The String Representation of LDAP Search Filters
- References for The String Representation of LDAP Search Filters
- Copyright Notice in The String Representation of LDAP Search Filters

A.6.1 Status of The String Representation of LDAP Search Filters

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements.

Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1997). All Rights Reserved.

A.6.2 IESG Note on The String Representation of LDAP Search Filters

This document describes a directory access protocol that provides both read and update access. Update access requires secure authentication, but this document does not mandate implementation of any satisfactory authentication mechanisms.

In accordance with RFC 2026, section 4.4.1, this specification is being approved by IESG as a Proposed Standard despite this limitation, for the following reasons:

a. to encourage implementation and interoperability testing of these protocols (with or without update access) before they are deployed, and

b. to encourage deployment and use of these protocols in read-only applications. (e.g. applications where LDAPv3 is used as a query language for directories which are updated by some secure mechanism other than LDAP), and

c. to avoid delaying the advancement and deployment of other Internet standards-track protocols which require the ability to query, but not update, LDAPv3 directory servers.

Readers are hereby warned that until mandatory authentication mechanisms are standardized, clients and servers written according to this specification which make use of update functionality are UNLIKELY TO INTEROPERATE, or MAY INTEROPERATE ONLY IF AUTHENTICATION IS REDUCED TO AN UNACCEPTABLY WEAK LEVEL.

Implementors are hereby discouraged from deploying LDAPv3 clients or servers which implement the update functionality, until a Proposed Standard for mandatory authentication in LDAPv3 has been approved and published as an RFC.

A.6.3 The String Representation of LDAP Search Filters Abstract

The Lightweight Directory Access Protocol (LDAP) [1] defines a network representation of a search filter transmitted to an LDAP server. Some applications may find it useful to have a common way of representing these search filters in a human-readable form. This document defines a human-readable string format for representing LDAP search filters.

This document replaces RFC 1960, extending the string LDAP filter definition to include support for LDAP version 3 extended match filters, and including support for representing the full range of possible LDAP search filters.

A.6.4 LDAP Search Filter Definition

Understand about the definition of an LDAP search filter.

An LDAPv3 search filter is defined in Section 4.5.1 of [1] as follows:

```
Filter::= CHOICE {
and [0] SET OF Filter,
or [1] SET OF Filter,
not [2] Filter,
equalityMatch [3] AttributeValueAssertion,
substrings [4] SubstringFilter,
greaterOrEqual [5] AttributeValueAssertion,
lessOrEqual [6] AttributeValueAssertion,
present [7] AttributeDescription,
approxMatch [8] AttributeValueAssertion,
extensibleMatch [9] MatchingRuleAssertion
}
SubstringFilter::= SEQUENCE {
type AttributeDescription,
SEQUENCE OF CHOICE {
initial [0] LDAPString,
any [1] LDAPString,
final [2] LDAPString
}
}
AttributeValueAssertion::= SEQUENCE {
attributeDesc AttributeDescription,
attributeValue AttributeValue
}
MatchingRuleAssertion::= SEQUENCE {
matchingRule [1] MatchingRuleID OPTIONAL,
type [2] AttributeDescription OPTIONAL,
matchValue [3] AssertionValue,
dnAttributes [4] BOOLEAN DEFAULT FALSE
```

}

AttributeDescription::= LDAPString

AttributeValue::= OCTET STRING

MatchingRuleID::= LDAPString

AssertionValue::= OCTET STRING

LDAPString::= OCTET STRING

where the LDAPString above is limited to the UTF-8 encoding of the ISO 10646 character set [4]. The AttributeDescription is a string representation of the attribute description and is defined in [1].

The AttributeValue and AssertionValue OCTET STRING have the form defined in [2]. The Filter is encoded for transmission over a network using the Basic Encoding Rules defined in [3], with simplifications described in [1].

A.6.5 String Search Filter Definition

Understand about the string representation of an LDAP search filter.

The string representation of an LDAP search filter is defined by the following grammar, following the ABNF notation defined in [5]. The filter format uses a prefix notation.

filter = "(" filtercomp ")"

filtercomp = and / or / not / item

and = "&" filterlist

or = "|" filterlist

not = "!" filter

filterlist = 1*filter

item = simple / present / substring / extensible

simple = attr filtertype value

filtertype = equal / approx / greater / less

equal = "="

approx = "~="

greater = ">="

less = "<="

extensible = attr [":dn"] [":" matchingrule] ":=" value

/ [":dn"] ":" matchingrule ":=" value

present = attr "=*"

substring = attr "=" [initial] any [final]

initial = value

any = "*" *(value "*")

final = value

attr = AttributeDescription from Section 4.1.5 of [1]

matchingrule = MatchingRuleId from Section 4.1.9 of [1]

value = AttributeValue from Section 4.1.6 of [1]

The attr, matchingrule, and value constructs are as described in the corresponding section of [1] given above.

If a value should contain any of the following characters

Character ASCII value ----------

* 0x2a

(0x28

) 0x29

\ 0x5c

NUL 0x00

then the character must be encoded as the backslash '\' character (ASCII 0x5c) followed by the two hexadecimal digits representing the ASCII value of the encoded character. The case of the two hexadecimal digits is not significant.

This simple escaping mechanism eliminates filter-parsing ambiguities and allows any filter that can be represented in LDAP to be represented as a NUL-terminated string. Other characters besides the ones listed above may be escaped using this mechanism, for example, non-printing characters.

For example, the filter checking whether the "cn" attribute contained a value with the character "*" anywhere in it would be represented as

"(cn=*\2a*)".

Note that although both the substring and present productions in the grammar above can produce the "attr=*" construct, this construct is used only to denote a presence filter.

A.6.6 Using String Search Filters

Understand how to use string search filters using few examples.

This section contains the following topics:

- Using Simple String Search Filters
- Using String Search Filters with Extensible Matching
- Using String Search Filters with Escaping Mechanism

A.6.6.1 Using Simple String Search Filters

The following examples illustrate simple use of string search filters:

(cn=Babs Jensen)

(!(cn=Tim Howes))

(&(objectClass=Person)(|(sn=Jensen)(cn=Babs J*)))

(o=univ*of*mich*)

A.6.6.2 Using String Search Filters with Extensible Matching

The following examples illustrate the use of extensible matching.

(cn:1.2.3.4.5:=Fred Flintstone)

(sn:dn:2.4.6.8.10:=Barney Rubble)

(o:dn:=Ace Industry)

(:dn:2.4.6.8.10:=Dino)

The second example illustrates the use of the ":dn" notation to indicate that matching rule "2.4.6.8.10" should be used when making comparisons, and that the attributes of an entry's distinguished name should be considered part of the entry when evaluating the match.

The third example denotes an equality match, except that DN components should be considered part of the entry when doing the match.

The fourth example is a filter that should be applied to any attribute supporting the matching rule given (since the attr has been left off). Attributes supporting the matching rule contained in the DN should also be considered.

A.6.6.3 Using String Search Filters with Escaping Mechanism

The following examples illustrate the use of the escaping mechanism.

(o=Parens R Us \28for all your parenthetical needs\29)

(cn=*\2A*)

(filename=C:\5cMyFile)

(bin=\00\00\00\04)

(sn=Lu\c4\8di\c4\87)

The first example shows the use of the escaping mechanism to represent parenthesis characters. The second shows how to represent a "*" in a value, preventing it from being interpreted as a substring indicator. The third illustrates the escaping of the backslash character.

The fourth example shows a filter searching for the four-byte value 0x00000004, illustrating the use of the escaping mechanism to represent arbitrary data, including NUL characters.

The final example illustrates the use of the escaping mechanism to represent various non-ASCII UTF-8 characters.

A.6.7 Security Considerations in The String Representation of LDAP Search Filters

This memo describes a string representation of LDAP search filters. While the representation itself has no known security implications, LDAP search filters do. They are interpreted by LDAP servers to select entries from which data is retrieved. LDAP servers should take care to protect the data they maintain from unauthorized access.

A.6.8 References for The String Representation of LDAP Search Filters

You can find references for the string representation of LDAP search filters in this section.

The following items are used as references:

[1] Wahl, M., Howes, T., and S. Kille, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[2] Wahl, M., Coulbeck, A., Howes, T., and S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

[3] Specification of ASN.1 encoding rules: Basic, Canonical, and Distinguished Encoding Rules, ITU-T Recommendation X.690, 1994.

[4] Yergeau, F., "UTF-8, a transformation format of Unicode and ISO 10646", RFC 2044, October 1996.

[5] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

A.6.9 Address of The String Representation of LDAP Search Filters Author

You can find the address of the string representation of LDAP search filters author.

Tim Howes Netscape Communications Corp. 501 E. Middlefield Road Mountain View, CA 94043 USA Phone: +1 415 937-3419 EMail: howes@netscape.com

A.6.10 Copyright Notice in The String Representation of LDAP Search Filters

Copyright (C) The Internet Society (1997). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other

Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY

RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

A.7 The Access Control Directive Format

The format (syntax) of any access control item (ACI) directive is defined by the user attribute `orclACI` and `orclEntryLevelACI`.
This appendix includes the following sections:

- Schema for orclACI
- Schema for orclEntryLevelACI

A.7.1 Schema for orclACI

The access control directive defined by the user attribute `orclACI` contains the schema elements `object identifier` and `accessDirectiveDescription`.

```
OrclACI:{ object_identifier NAME 'orclACI' DESC 'Stores an inheritable ACI' EQUALITY
accessDirectiveMatch SYNTAX 'accessDirectiveDescription'  USAGE
'directoryOperation'}

accessDirectiveDescription has the following BNF:
<accessDirectiveDescription>
                  ::= access to <object> [by <subject> ( <accessList> )]+

<object> ::= [attr <EQ-OR-NEQ> ( * | (<attrList>) ) | entry]
[filter=(<ldapFilter>)] [DenyGroupOverride] [AppendToAll]

<subject> ::= <entity> [<BindMode>] [<BindIPFilter>]
[Added_object_constraint=(<ldapFilter>)]
<entity> ::= * | self | dn="<regex>" | dnAttr=(<dn_attribute>) | group="<dn>" |
guidattr=(<guid_attribute>) | groupattr=(<group_attribute>) | [SuperUser]

BindMode=(LDAP_authentication_choice)|LDAP_security_choice)
LDAP_authentication_choice::= proxy | simple | MD5Digest | PKCS12
LDAP_security_choice::= SSLNoAuth | SSLOneWay | SASL

BindIPFilter=(<ldapFilter for orclipaddress>)
ex: (|(orclipaddress=1.2.3.*)(orclipaddress=1.2.4.*)), (&(orclipaddress!=1.2.*)
(orclipaddress!=3.4.*))

<accessList> ::= <access> | <access>, <accessList>

<access> ::= none | compare | search | browse | proxy | read | selfwrite | write |
add | delete | nocompare | nosearch | nobrowse | noproxy |noread | noselfwrite |
nowrite | noadd | nodelete

<attrList> ::=  <attribute name> | <attribute name>,<attrList>

<EQ-OR-NEQ> ::=  = | !=

<regex> ::= <dn> | *,<dn_of_any_subtree_root>
```

> **Note:**
>
> The regular expression defined earlier is not meant to match any arbitrary expression. The syntax only allows expressions where the wildcard is followed by a comma and a valid DN. The latter DN denoted by *<dn_of_any_subtree_root>* is intended to specify the root of some subtree.

A.7.2 Schema for orclEntryLevelACI

The BER format for `orclEntryLevelACI` is the same as the format for `orclACI`.

The entry level access control directive defined by the user attribute `orclEntryLevelACI` has the following schema:

```
"orclEntryLevelACI":
{ object_identifier NAME 'orclEntryLevelACI' DESC 'Stores entry level ACL Directive'
EQUALITY accessDirectiveMatch SYNTAX 'orclEntryLevelACIDescription'
USAGE 'directoryOperation' }

<orclEntryLevelACIDescription>
::= access to <object> [by <subject> ( <accessList> )]+
```

A.8 Globalization Support in the Directory

Oracle Internet Directory uses Globalization Support to store, process and retrieve data in native languages. It ensures that Oracle Internet Directory utilities and error messages automatically adapt to the native language and locale.
This section describes globalization support as used by Oracle Internet Directory, including the required `NLS_LANG` environment variables for the various components and tools in an Oracle Internet Directory environment.

> **See Also:**
>
> Globalization Support before configuring Globalization Support.

This appendix includes the following sections:

- About Character Sets and the Directory
- Components of the NLS_LANG Parameter
- Limitation of using Non-AL32UTF8 Databases
- Using Globalization Support with LDIF Files
- Using Globalization Support with Command-Line LDAP Tools
- Setting NLS_LANG in the Client Environment
- Using Globalization Support with Bulk Tools

A.8.1 About Character Sets and the Directory

When computer systems process characters, they use numeric codes instead of the graphical representation of the character. For example, when the database stores the letter A, it actually stores a numeric code that is interpreted by software as the letter.

A group of characters (for example, alphabetic characters, ideographs, symbols, punctuation marks, and control characters) can be encoded as a character set. Each encoded character set assigns a unique code to each character in the set. For example, in the ASCII encoding scheme, the character code of the first character of the English upper-case alphabet is Ox4; in the EBCDIC encoding scheme, it is Oxc1.

The computer industry uses many encoded character sets. These character sets can differ in the number and types of characters available and in many other ways as well.

When you create a database, you specify an encoded character set. Choosing a character set determines, among other things, what languages can be represented in the database.

Oracle supports most national, international, and vendor-specific encoded character set standards.

This section contains the following topics:

- About Unicode
- Unicode Implementations
- UTF-8 Support in Oracle Databases

A.8.1.1 About Unicode

No single character set contains enough characters to meet the requirements of day-to-day e-business requirements. For example, no one national character set can represent all the languages in the European Union. Moreover, there are potential conflicts between character sets because the same character can be represented by different codes in different character sets.

To overcome these obstacles, a global character set, called Unicode, was developed. It is a universal encoded character set that can store information from any language including punctuation marks, diacritics, mathematical symbols, technical symbols, musical symbols, and so forth. As of version 3.2, the Unicode Standard supports over 95,000 characters from the world's alphabets, ideograph sets, and symbol collections. It includes 45,000 supplementary characters, most of which are Chinese, Japanese, and Korean characters that are rarely used but nevertheless need representation in electronic documentation.

A.8.1.2 Unicode Implementations

Unicode has more than one implementation standard, and these are described in Table A-24.

Table A-24 Unicode Implementations

Implementation	Description
UTF-8	A variable-width 8-bit encoding of Unicode. One Unicode character can be one, two, three, or four bytes. Characters from European scripts are represented in one or two bytes. Those from Asian scripts are represented in three bytes, and supplementary characters are represented in four.

Table A-24 (Cont.) Unicode Implementations

Implementation	Description
UCS-2	A fixed-width 16-bit encoding of Unicode in which each character, regardless of the script, is two bytes.
UTF-16	The 16-bit encoding of Unicode. It is an extension of UCS-2 that supports the supplementary characters added in Unicode 3.1. One character can be two or four bytes. Characters from European and Asian scripts are represented in two bytes, and supplementary characters are represented in four.

A.8.1.3 UTF-8 Support in Oracle Databases

Oracle began supporting Unicode as a database character set beginning with Oracle database version 7. With Oracle9*i*, Oracle added a new UTF-8 character set called AL32UTF8. This database character set supports the latest version of Unicode (3.2), including the latest supplementary characters. Oracle intends to enhance AL32UTF8 as necessary to support future versions of the Unicode standard.

A.8.2 Components of the NLS_LANG Parameter

Each component of NLS_LANG parameter controls the operation of a subset of Globalization Support features.

The `NLS_LANG` parameter has three components—`language`, `territory`, and `charset`—in the form:

```
NLS_LANG = language_territory.charset
```

Components of the NLS_LANG parameter are shown in Table A-25.

Table A-25 Components of the NLS_LANG Parameter

Component	Description
`language`	Specifies conventions such as the language used for Oracle messages, day names, and month names. Each supported language has a unique name—for example, American English, French, or German. If language is not specified, the value defaults to American English. **See Also:** NLS_LANGUAGE in *Oracle Database Globalization Support Guide* in the Oracle Database Documentation Library for a complete list of languages
`territory`	Specifies conventions such as the default calendar, collation, date, monetary, and numeric formats. Each supported territory has a unique name; for example, America, France, or Canada. If territory is not specified, the value defaults to America. **See Also:** NLS_TERRITORY in *Oracle Database Globalization Support Guide* in the Oracle Database Documentation Library for a complete list of territories

Table A-25 (Cont.) Components of the NLS_LANG Parameter

Component	Description
`charset`	Specifies the character set used by the client application (normally that of the user's terminal). Each supported character set has a unique acronym, for example, WE8MSWIN1252, JA16SJIS, or AL32UTF8. **See Also:** in *Oracle Database Globalization Support Guide* in the Oracle Database Documentation Library for a complete list of character sets

A.8.3 Setting NLS_LANG Parameter from the Command Line

You can set `NLS_LANG` as an environment variable at the command line.

The following are examples of legal values for `NLS_LANG`:

- `AMERICAN_AMERICA.AL32UTF8`
- `JAPANESE_JAPAN.AL32UTF8`

A.8.4 Limitation of using Non-AL32UTF8 Databases

You can run the Oracle directory server and database tools on a non-AL32UTF8 database, but be sure that all characters in the client character set are included in the database character set (with the same or different codes).

Otherwise, you can lose data during ldapadd, ldapdelete, ldapmodify, or ldapmodifydn operations. For example, suppose that you perform an ldapadd operation using a multibyte character set on an underlying database that uses only single-byte characters. You will lose data because not all of the bytes you enter are accepted by the database.

A.8.5 Using Globalization Support with LDIF Files

Attribute type names are always ASCII strings that cannot contain multibyte characters. Oracle Internet Directory does not support multibyte characters in attribute type names. However, Oracle Internet Directory does support attribute *values* containing multibyte characters such as those in the simplified Chinese (ZHS16GBK) character set.

Attribute values can be encoded in different ways to allow Oracle Internet Directory tools to interpret them properly. There are two scenarios, described the following sections:

- Interpretation of LDIF file Containing Only ASCII Strings
- Interpreting LDIF file Containing UTF-8 Encoded Strings

> **See Also:**
>
> LDIF File Formatting in *Reference for Oracle Identity Management*.

A.8.5.1 Interpretation of LDIF file Containing Only ASCII Strings

In this scenario, character strings for attribute values are also in ASCII.

Because all tools use the UTF-8 character set by default, and ASCII is a proper subset of UTF-8, all tools can interpret these files. The same is true of keyboard input of values that are simply ASCII strings.

A.8.5.2 Interpreting LDIF file Containing UTF-8 Encoded Strings

In this scenario, character strings for attribute values are also in UTF-8.

Because, by default, all tools use the UTF-8 character set, all tools can interpret these files. The same is true of keyboard input of values that are UTF-8 strings.

In such a file, some characters may be multibyte. Multibyte characters strings can be present in the LDIF files as attribute values or given as keyboard input. They can be encoded in their native character set or in UTF-8. They can also be BASE64 encoded representations of either the native or the UTF-8 string.

Consider the following cases:

- Using Non-UTF-8 Native Strings
- Using UTF-8 Strings
- Using BASE64 Encoded UTF-8 Strings
- Using BASE64 Encoded Native Strings

Because the directory server understands and expects only UTF-8 encoded strings, the first, third, and fourth cases need to undergo conversion to UTF-8 strings before they can be sent to the LDAP server.

A.8.5.2.1 Using Non-UTF-8 Native Strings

If `NLS_LANG` is not set, use the `-E` *`character_set`* argument with the command-line LDAP tools `ldapadd`, `ldapaddmt`, `ldapbind`, `ldapcompare`, `ldapmoddn`, `ldapmod`, `ldapdelete`, and `ldapsearch`. You do not need to use the `-E` *`character_set`* argument if `NLS_LANG` is set.

This example converts simplified Chinese native strings to UTF-8. The baseDN can be a simplified Chinese string:

```
ldapsearch -h my_host -D "cn=orcladmin" -q -p 3060 -E ".ZHS16GBK" -b base_DN \
   -s base  "objectclass=*"
```

Use the `encode="`*`character_set`*`"` argument with the command-line bulk tools `bulkload`, `bulkmodify`, `bulkdelete`, and `ldifwrite`, where *`character_set`* is the character set used in the LDIF file. Set `NLS_LANG` to the character set used by the directory's database.

A.8.5.2.2 Using UTF-8 Strings

No conversion is required.

A.8.5.2.3 Using BASE64 Encoded UTF-8 Strings

You do not need to use the `-E` *`character_set`* argument with the LDAP tools, even if `NLS_LANG` is not set.

You do not need to use the `encode=`*`character_set`* argument with the command-line tools. Oracle Internet Directory tools automatically decode BASE64 encoded UTF-8 strings to UTF-8 strings.

A.8.5.2.4 Using BASE64 Encoded Native Strings

If `NLS_LANG` is not set, use the `-E character_set` argument with the command-line LDAP tools `ldapadd`, `ldapaddmt`, `ldapbind`, `ldapcompare`, `ldapmoddn`, `ldapmod`, `ldapdelete`, and `ldapsearch`. You do not need to use the `-E character_set` argument if `NLS_LANG` is set.

Use the `encode="character_set"` argument with the command-line bulk tools `bulkload`, `bulkmodify`, `bulkdelete`, and `ldifwrite`, where `character_set` is the character set used in the LDIF file. Set `NLS_LANG` to the character set used by the directory's database.

Oracle Internet Directory tools automatically decode BASE64 encoded native strings to simple native strings. The native strings are then converted to the equivalent UTF-8 strings.

> **Note:**
>
> In any given input file, only one character set may be used.

A.8.6 Using Globalization Support with Command-Line LDAP Tools

The Oracle Internet Directory command-line tools read keyboard input or LDIF file.

The command-line tool read keyboard input in the following ways:

- ASCII characters only
- Non-ASCII input (native language character set)
- BASE64 encoded values of UTF-8 or native strings (from LDIF file only)

If the character set being given as input from an LDIF file or keyboard is not UTF-8, then the command-line tools need to convert the input into UTF-8 format before sending it to the LDAP server.

This section contains these topics:

- Enabling Command-line Tools to Convert Other Character Set Input to UTF-8
- Specifying the -E Argument When Using Each Tool
- Using -E Argument with Command-Line LDAP Tools

A.8.6.1 Enabling Command-line Tools to Convert Other Character Set Input to UTF-8

You enable the command-line tools to convert the input into UTF-8 by specifying the `-E character_set` argument with any of the command-line LDAP tools.

Use the `encode="character_set"` argument with `bulkload`, `bulkmodify`, `bulkdelete`, and `ldifwrite`.

A.8.6.2 Specifying the -E Argument When Using Each Tool

The client tools always assume UTF-8 (the Oracle character set name is AL32UTF8) to be the character set unless otherwise specified by the `-E` argument. The BASE64-encoded values are decoded, and then the decoded buffer is converted to UTF-8 if the `-E` argument is specified.

For example, if you specify `-E ".ZHS16GBK"`, then the decoded buffer is converted from simplified Chinese GBK to Unicode UTF-8 before being sent to the directory server.

Specifying the `-E` argument ensures that proper character set conversion can occur from the character set you specify for the `-E` argument (`-E ".`*character_set*`"`) to the AL32UTF8 character set.

The command-line tools use the `-E` argument to process the input in the character set specified for the `-E` argument. They display their output in the character set specified in the `NLS_LANG` environment variable.

For example, to add entries from an LDIF file encoded in the simplified Chinese character set (ZHS16GBK) by using ldapadd, type:

```
ldapadd -h myhost -p 3060 -E ".ZHS16GBK" -D cn=orcladmin -q -f my_ldif_file
```

In this example, the ldapadd tool converts the characters from `".ZHS16GBK"` (simplified Chinese character set) to `".AL32UTF8"` before they are sent across the wire to the directory server.

A.8.6.3 Using -E Argument with Command-Line LDAP Tools

You have to use proper -E Argument with Command-Line LDAP Tools.

Table A-26 provides additional examples of how to use the `-E` argument correctly for each command-line tool. In each example, the command converts data from simplified Chinese, as specified by the value `".ZHS16GBK"`, to AL32UTF8. For example, in each command, the value for the `-D` option and the password typed at the prompt when `-q` is specified are in GBK. Specifying the `-E` argument converts them to UTF-8.

Note that, in the examples in Table A-26, we do not show any actual characters belonging to the `.ZHS16GBK` character set. These examples would, therefore, work without the `-E` argument. However, if the argument values contained actual characters in the `.ZHS16GBK` character set, then we would need to use the `-E` argument.

> **See Also:**
>
> Oracle Internet Directory Administration Tools in *Reference for Oracle Identity Management* for syntax and usage notes for each of the command-line tools

Table A-26 Examples: Using the -E Argument with Command-Line Tools

Tool	Example
`ldapbind`	`ldapbind -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`
`ldapsearch`	`ldapsearch -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`

Table A-26 (Cont.) Examples: Using the -E Argument with Command-Line Tools

Tool	Example
ldapadd	`ldapadd -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`
ldapaddmt	`ldapaddmt -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`
ldapmodify	`ldapmodify -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`
ldapmodifymt	`ldapmodifymt -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`
ldapdelete	`ldapdelete -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" -q`

ldapcompare	`ldapcompare -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" \`
	`-b "ou=Construction,ou=Manufacturing,o=example,c=us" \`
	`-a title -v manager -q`

ldapmoddn	`ldapmoddn -h my_host -p 3060 -E ".ZHS16GBK" -D "o=example,c=us" \`
	`-b "cn=Frank Smith,ou=Construction,ou=Manufacturing,c=us, o=example"\`
	`-N "ou=Contracting,ou=Manufacturing,o=example,c=us" -r -q`

A.8.7 Setting NLS_LANG in the Client Environment

If the output required by the client is UTF-8, then you do not need to set the `NLS_LANG` environment variable. In this case, the character set component of the `NLS_LANG` environment variable defaults to AL32UTF8, and both the input path from client to server, and the output path from server to client, do not require any character set conversion.

If the output required by the client is *not* UTF-8, then you must set the `NLS_LANG` environment variable. This ensures that proper character set conversion can occur from the AL32UTF8 character set to the character set required by the client.

For example, if the `NLS_LANG` environment variable is set to the simplified Chinese character set, then the command-line tool displays output in that character set. Otherwise the output defaults to the AL32UTF8 character set.

> **Note:**
>
> If you are using Microsoft Windows, then, to use the command-line tools after server startup, you must reset `NLS_LANG` in an MS-DOS window. Set it to the character set that matches the code page of your MS-DOS session. AL32UTF8 cannot be used. See the *Oracle® Database Installation Guide for Microsoft Windows* for more information on which character set to use for command-line tools in an MS-DOS session.
>
> If you are using a pre-installed Oracle Database with Oracle Internet Directory, then you must also set the database character set to AL32UTF8.
>
> **See Also:** NLS_LANGUAGE in *Oracle Database Globalization Support Guide* in the Oracle Database Documentation Library and *Oracle Database Installation Guide for Microsoft Windows*.
>
> Be careful not to change the `NLS_LANG` parameter value in the registry.

A.8.8 Using Globalization Support with Bulk Tools

Oracle Internet Directory ensures that the reading and writing of text data from and to LDIF files are done in UTF-8 encoding as specified by the LDAP standard.

This section provides examples of the argument you use for each of the bulk tools. It contains the following sections:

- Using Globalization Support with bulkload
- Using Globalization Support with ldifwrite
- Using Globalization Support with bulkdelete
- Using Globalization Support with bulkmodify

> **See Also:**
>
> Oracle Internet Directory Administration Tools in *Reference for Oracle Identity Management* for a list of arguments for each bulk tool

A.8.8.1 Using Globalization Support with bulkload

Add to the command the argument `encode="`*`character_set`*`"` where the input LDIF file is encoded in *"character_set"*.

For example, ensure that DOMAIN_HOME is set, then type:

```
bulkload connect="connect_string" \
   encode=".ZHS16GBK" check="TRUE" generate="TRUE" file="my_ldif_file"
```

A.8.8.2 Using Globalization Support with ldifwrite

The ldifwrite utility always writes BASE64 encoded values for multibyte strings.

The BASE64 encoding could be of the UTF-8 strings as they are stored in the directory server, or of native strings as specified by the `NLS_LANG` environment variable setting when running ldifwrite.

For example:

```
ldifwrite connect="connect_string" baseDN="baseDN" ldiffile="output_file"
```

In this example, if the `NLS_LANG` environment variable is not set, or is set to *`language_territory`*`.AL32UTF8`, then the output LDIF file will contain BASE64-encoded UTF-8 strings for any multibyte characters.

For information about loading this LDIF file into a directory, see "Using BASE64 Encoded UTF-8 Strings".

If the `NLS_LANG` environment variable is set to a character set other than AL32UTF8—for example, `".ZHS16GBK"`—then the output LDIF file will contain a BASE64 encoded value of simplified Chinese GBK strings.

For information about loading this LDIF file into the directory, see "Using BASE64 Encoded Native Strings".

A.8.8.3 Using Globalization Support with bulkdelete

Add `encode="`*`character_set`*`"` to the command.

For example:

```
bulkdelete connect="connect_string"  encode=".ZHS16GBK" \
    baseDN="ou=manufacturing,o=example,c=us"
```

In this case the value for the `-base` option could be in the `ZHS16GBK` native character set, that is, simplified Chinese.

A.8.8.4 Using Globalization Support with bulkmodify

Add `encode="`*`character_set`*`"` to the command the argument.

For example:

```
bulkmodify connect="my_service_name" \
            encode=".ZHS16GBK" baseDN="ou=manufacturing,o=example,c=us" \
            replace=TRUE" value=Foreman filter="objectclass=*"
```

In this example, values for the `basedn`, `value`, and `filter arguments` can be specified using the simplified Chinese GBK character set.

A.9 Setting up Access Controls for Creation and Search Bases for Users and Groups

You can set up access controls in Oracle Internet Directory for the User Search Base, User Creation Base, Group Search Base, and Group Creation Base.

> **Note:**
>
> If you modify the User Search Base, the User Creation Base, the Group Search Base, or the Group Creation Base, then access controls for the new container need to be set up properly.

It includes the following sections:

- Setting up Access Controls for the User Search Base and the User Creation Base
- Setting up Access Controls for the Group Search Base and the Group Creation Base

A.9.1 Setting up Access Controls for the User Search Base and the User Creation Base

To set up access controls for the User Search Base and the User Creation Base you need to create an LDIF file.

Perform the following steps:

1. Create an LDIF (user_aci.ldif) file with the following entry:

```
--- BEGIN LDIF file contents---
dn: %usersearch_or_createbase_dn%
changetype: modify
add: orclaci
orclaci: access to entry by group="cn=oracledascreateuser,
 cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orcluser*) (browse,add) by
 group="cn=Common User Attributes, cn=Groups,
 cn=OracleContext,%subscriberdn%" (browse) by
 group="cn=PKIAdmins, cn=groups, cn=OracleContext,%subscriberdn%" (browse)
orclaci: access to entry filter=(objectclass=inetorgperson) by
 group="cn=oracledascreateuser, cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orcluser*) (browse,add) by
 group="cn=oracledasdeleteuser, cn=groups,cn=OracleContext,%subscriberdn%"
 (browse,delete) by group="cn=oracledasedituser,
 cn=groups,cn=OracleContext,%subscriberdn%" (browse) by
 group="cn=UserProxyPrivilege, cn=Groups,cn=OracleContext,%subscriberdn%"
 (browse,
 proxy) by dn="orclApplicationCommonName=DASApp, cn=DAS,
 cn=Products,cn=oraclecontext" (browse,proxy) by self (browse, nodelete, noadd)
 by
 group="cn=Common User Attributes, cn=Groups,cn=OracleContext,%subscriberdn%"
 (browse) by * (browse, noadd, nodelete)
orclaci: access to attr=(*) filter=(objectclass=inetorgperson) by
 group="cn=oracledasedituser, cn=groups,cn=OracleContext,
 %subscriberdn%" (read,search,write,compare) by self (
 read,search,write,selfwrite,compare) by *
 (read, nowrite, nocompare)
orclaci: access to attr=(userPassword)
 filter=(objectclass=inetorgperson) by
 group="cn=OracleUserSecurityAdmins,cn=Groups,
 cn=OracleContext, %subscriberdn%"
 (read,search,write,compare) by group="cn=oracledasedituser,
 cn=groups,cn=OracleContext,%subscriberdn%"
 (read,search,write,compare) by self
 (read,search,write,selfwrite,compare) by group="cn=authenticationServices,
```

```
 cn=Groups,cn=OracleContext,%subscriberdn%" (compare) by * (none)
orclaci: access to attr=(authpassword, orclpasswordverifier, orclpassword) by
 group="cn=oracledasedituser,cn=groups,cn=OracleContext,%subscriberdn%"
 (read,search,write,compare) by
 group="cn=verifierServices,cn=Groups,cn=OracleContext,%subscriberdn%"
 (search, read, compare) by self (search,read,write,compare) by * (none)
orclaci: access to attr=(orclpwdaccountunlock) by
 group="cn=oracledasedituser,cn=groups,cn=OracleContext,%subscriberdn%" (
 write) by * (none)
orclaci: access to attr=(usercertificate, usersmimecertificate) by
 group="cn=PKIAdmins,cn=Groups,cn=OracleContext,%subscriberdn%"
 (read, search, write, compare) by self (read, search, compare) by *
 (read, search, compare)
orclaci: access to attr=(mail) by
 group="cn=EmailAdminsGroup,cn=EmailServerContainer,cn=Products,
 cn=OracleContext" (write) by group="cn=oracledasedituser,
 cn=groups,cn=OracleContext,%subscriberdn%" (read,search,write,compare)
orclaci: access to attr=(orclguid, orclisenabled, modifytimestamp,mail)
 by group="cn=Common User Attributes,
 cn=Groups,cn=OracleContext,%subscriberdn%"
 (read, search, compare) by group="cn=oracledasedituser,
 cn=groups,cn=OracleContext,%subscriberdn%" (read,search,write,compare)
 by * (read, nowrite, nocompare)
orclaci: access to attr=(orclpasswordhintanswer) by
 group="cn=Common User Attributes,
 cn=Groups,cn=OracleContext,%subscriberdn%" (read, search, compare) by self
 (read,search,write,selfwrite,compare) by * (noread, nowrite, nocompare)
orclaci: access to attr=(orclpasswordhint) by
 group="cn=Common User Attributes,
 cn=Groups,cn=OracleContext,%subscriberdn%" (read, search, compare) by self
 (read,search,write,selfwrite,compare) by
 group="cn=OracleUserSecurityAdmins,cn=Groups,cn=OracleContext,
 %subscriberdn%" (read,search,write,compare) by *
 (noread, nowrite, nocompare)
orclaci: access to attr=(displayName, preferredlanguage,
 orcltimezone,orcldateofbirth,orclgender,orclwirelessaccountnumber,cn,
 uid,homephone,telephonenumber) by group="cn=Common User Attributes,
 cn=Groups,cn=OracleContext,%subscriberdn%"
 (read, search, compare) by group="cn=oracledasedituser,
 cn=groups,cn=OracleContext,%subscriberdn%" (read,search,write,compare)
 by self (read,search,write,selfwrite,compare) by *
 (read, nowrite, nocompare)
        -
add: orclentrylevelaci
orclentrylevelaci: access to entry by group="cn=oracledascreateuser,
 cn=groups,cn=OracleContext,%subscriberdn%" added_object_constraint=
 (objectclass=orcluser*) (browse, add) by * (browse)
---END LDIF file contents------
```

2. Replace `%subscriberdn%` with the dn of the subscriber and `%usersearch_or_createbase_dn%` with the new value of the container DN where the new user search/create base points to.
3. Run the `ldapmodify` command as follows:

```
ldapmodify -p oidport -h oidhost -D cn=orcladmin -q -v \
           -f  user_aci.ldif
```

A.9.2 Setting up Access Controls for the Group Search Base and the Group Creation Base

To set up access controls for the Group Search Base and the Group Creation Base, create an LDIF file as the first step.

Perform the following steps:

1. Create an LDIF (group_aci.ldif) file with the following entry:

```
--- BEGIN LDIF file contents---
dn: %groupsearch_or_createbase_dn%
changetype: modify
add: orclaci
orclaci: access to entry by group="cn=IASAdmins,
 cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orclcontainer) (browse,add)
orclaci: access to entry by group="cn=oracledascreategroup,
 cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orclgroup*) (browse,add) by
 group="cn=Common
 Group Attributes, cn=Groups,cn=OracleContext,%subscriberdn%" (browse)
orclaci: access to entry filter=(&(objectclass=orclgroup)(orclisvisible=false))
 by
 groupattr=(owner) (browse, add, delete) by dnattr=(owner)
 (browse, add, delete) by
 group="cn=Common Group Attributes, cn=Groups,cn=OracleContext,%subscriberdn%"
 (browse) by * (none)
orclaci: access to entry
 filter=(&(objectclass=orclgroup)(!(orclisvisible=false))) by
 group="cn=oracledascreategroup, cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orclgroup) (browse,add) by
 group="cn=oracledasdeletegroup, cn=groups,cn=OracleContext,%subscriberdn%"
 (browse,delete) by group="cn=oracledaseditgroup,
 cn=Groups,cn=OracleContext,%subscriberdn%" (browse) by groupattr=(owner) (
 browse,
 add, delete) by dnattr=(owner) (browse, add, delete) by group="cn=Common Group
 Attributes, cn=Groups,cn=OracleContext,%subscriberdn%" (browse)
orclaci: access to attr=(*)
 filter=(&(objectclass=orclgroup)(orclisvisible=false)) by
 groupattr=(owner) (read,search,write,compare) by dnattr=(owner)
 (read,search,write,compare) by * (none) by group="cn=Common Group Attributes,
 cn=Groups,cn=OracleContext,%subscriberdn%" (read, search, compare)
orclaci: access to attr=(*)
 filter=(&(objectclass=orclgroup)(!(orclisvisible=false))) by
 groupattr=(owner) (read,search,write,compare) by dnattr=(owner)
 (read,search,write,compare)  by group="cn=oracledaseditgroup,
 cn=groups,cn=OracleContext,%subscriberdn%" (read,search,write,compare) by
 group="cn=Common Group Attributes, cn=Groups,cn=OracleContext,%subscriberdn%"
 (read, search, compare)
      -
add: orclentrylevelaci
orclentrylevelaci: access to entry by group="cn=oracledascreategroup,
 cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orclgroup) (browse, add) by
 group="cn=IASAdmins, cn=groups,cn=OracleContext,%subscriberdn%"
 added_object_constraint=(objectclass=orclcontainer) (browse,add) by * (browse)
---END LDIF file contents------
```

2. Replace `%subscriberdn%` with the DN of the subscriber and `%groupsearch_or_createbase_dn%` with the new value of the container DN where the new group search base or group create base points to.
3. Run the ldapmodify command as follows:

```
ldapmodify -p oidport -h oidhost -D cn=orcladmin -q -v -f group_aci.ldif
```

A.10 Searching the Directory for User Certificates

You can search Oracle Internet Directory for user certificates by certificate mapping and certificate matching methods.
You can perform a command-line search of the binary attribute `usercertificate`.

Before 10*g* Release 2 (10.1.2.0.2), the only way to identify a user from the certificate was through the DN specified in the certificate. This is known as certificate matching. Oracle Internet Directory now supports certificate mapping, in addition to certificate matching. Certificate matching requires that a user certificate be provisioned in the directory. Certificate mapping does not require provisioning of a user certificate.

It includes the following sections:

- Mapping the Certificate
- Search Types

A.10.1 Mapping the Certificate

Certificate mapping allows a customer to define rules for mapping the certificate to the user's DN. A certificate mapping rule is a set of rules for parsing the certificate and for querying the directory for the user's identity. Only custom extensions of certificates can be used in mapping rules.

You can perform the following tasks:

- Adding a Certificate Mapping Rule
- Deleting a Certificate Mapping Rule
- Modifying a Certificate Mapping Rule

A.10.1.1 Adding a Certificate Mapping Rule

Use the `ldapmodify` command to add the mapping rule as follows:

```
ldapmodify -D "cn=orcladmin" -q -h hostName -p port_number -f certMapRuleAdd.ldif
```

The file `certMapRuleAdd.ldif` should look something like this:

```
dn: cn=maprule1,cn=SASL-EXTERNAL,cn=Identity Mapping Configurations,cn=Server
Configurations
cn: maprule1
objectclass: orclidmapping
objectclass: orclcertidmapping
orclSearchScope: subtree
orclSearchFilter: (cn=$(2.16.750.5.14.2.81.2.5.1))
orclcertExtensionOID: 2.16.750.5.14.2.81.2.5
orclcertExtensionAttribute: 2.16.750.5.14.2.81.2.5.1
```

A.10.1.2 Deleting a Certificate Mapping Rule

Use the `ldapdelete` command to delete the mapping rule as follows:

```
ldapdelete hostName -D "cn=orcladmin" -q -p port_number \
  "cn=maprule1,cn=SASL-EXTERNAL,cn=Identity Mapping Configurations,cn=Server \
  Configurations"
```

A.10.1.3 Modifying a Certificate Mapping Rule

Use the `ldapmodify` command to modify the mapping rule as follows:

```
ldapmodify -D "cn=orcladmin" -q -h hostName -p port_number -f certMapRuleMod.ldif
```

The file `certMapRuleMod.ldif` should look something like this:

```
dn: cn=maprule1,cn=SASL-EXTERNAL,cn=Identity Mapping Configurations,cn=Server
Configurations
changetype:modify
replace: attrName
attrName: attrValue
```

A.10.2 Search Types

You can use two kinds of certificate search filters.

The two types of certificate search filters are:

- A filter of the form "`usercertificate=certificate_serial_number$certificate_issuer_DN`". A combination of the certificate serial number and the certificate issuer's DN is used to locate the certificate. This combination is called the certificate match value.
- A filter of the form"`usercertificate;binary=base_64_encoded_value_of_certificate`". Using this filter, one of six types of searches is possible, depending upon two things:
  - The value of the DSA configuration set attribute (`DN`: "`cn=dsaconfig,cn=configsets,cn=oracle internet directory`"), `orclpkimatchingrule`.
  - The presence or absence of the LDAP control `GSL_CERTIFICATE_CONTROL`, 2.16.840.1.113894.1.8.23

  The six types of searches possible with a filter of the form "`usercertificate;binary=base_64_encoded_value_of_certificate`" are:

Presence of LDAP control	Value of orclpkimatchingrule	Search Behavior
Absent	Not used	The hashed value of the client certificate is used to locate `usercertificate`.
Present	0	An exact-match search is performed. The subject DN of the client certificate is the search base. This DN is compared with the user DN in the directory. The search scope is `Base`. The filter is "`objectclass=*`".
Present	1	The hashed value of the client certificate is used to locate `usercertificate`.

Presence of LDAP control	Value of orclpkimatchingrule	Search Behavior
Present	2 (Default)	The hashed value of the client certificate is used to locate `usercertificate`. If this search yields nothing, An exact-match search is performed.
Present	3	The mapping rule is used.
Present	4	First, the mapping rule is used. If that search yields nothing, then the search proceeds as if the value were 2.

For information on using LDAP controls, see Extensions to the LDAP Protocol in *Oracle Fusion Middleware Application Developer's Guide for Oracle Identity Management*.

Note:

- The `usercertificate` attribute cannot be searched using a substring filter.
- In an exact-match search, the search filter can contain only one attribute value assertion.
- Only one-level and subtree searches are supported.
- The `catalog` tool does not support catalogs for user certificates—namely ct_orclcertificatehash and ct_orclcertificatematch
- The certificate hash values requires that certificates be upgraded from earlier releases. See `upgradecert.pl` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

See Also:

"About Direct Authentication"

A.11 Adding a Directory Node by Using the Database Copy Procedure

You need to add a new node to an existing replicating system for Oracle Internet Directory by using the database copy procedure, also known as cold backup.

Note:

This procedure works only for Oracle Internet Directory. Do not use this procedure if other Oracle Identity Management components such as Oracle Single Sign-On are installed. You can use the database copy procedure to create a new directory replication group (DRG) if you have a standalone Oracle Internet Directory node. This procedure is applicable for a full LDAP-based replica.

The following sections explain it further:

- Definition of Sponsor Site and New Site in Database Copy Procedure
- Prerequisites for Database Copy Procedure
- Sponsor Directory Site Environment for Database Copy Procedure
- New Directory Site Environment for Database Copy Procedure
- Adding a Directory Node by using Database Copy Procedure

A.11.1 Definition of Sponsor Site and New Site in Database Copy Procedure

The sponsor site refers to the site or host or node where Oracle Internet Directory and its repository, the Oracle database, are installed.

The sponsor site is also referred to as sponsor node.

The new site refers to the site or host or node to which you are copying the Oracle Internet Directory repository. The new site is also referred to as the new node.

A.11.2 Prerequisites for Database Copy Procedure

Your computing environment must meet certain prerequisites before database copy procedure,

Prerequisites before you start this procedure:

- The operating system, version, and patch level of the new directory site must be the same as that of the sponsor directory site. This procedure might not work if the patch levels of the operating systems differ.
- Oracle strongly recommends that you back up the sponsor directory's repository before you employ this procedure.
- Because this procedure involves copying Oracle data files, performance depends on the underlying network. If the underlying network is slow, consider using one of the methods described in Setting Up Replication to set up a replication group. Alternatively, you could physically transfer compressed Oracle data files on removable media. Consult your local system or network administrator for information about the network.
- Only a person familiar with the Oracle database should perform this procedure.

A.11.3 Sponsor Directory Site Environment for Database Copy Procedure

The sponsor directory site environment for database copy procedure is explained with an example.

In the example shown throughout this chapter, the sponsor directory site's environment is as follows:

```
Hostname = rst-sun
Domain name = example.com
ORACLE_BASE = /private/oracle/app/oracle
ORACLE_HOME = /private/oracle/app/oracle/product/OraHome_1
ORACLE_SID  = LDAP
LD_LIBRARY_PATH = $ORACLE_HOME/lib
NLS_LANG = AMERICAN_AMERICA.AL32UTF8
datafile location = /private/oracle/oradata/LDAP
Dump destination =  /private/oracle/app/oracle/admin/LDAP/pfile,
                    /private/oracle/app/oracle/admin/LDAP/bdump,
                    /private/oracle/app/oracle/admin/LDAP/cdump,
```

```
                    /private/oracle/app/oracle/admin/LDAP/udump,
                    /private/oracle/app/oracle/admin/LDAP/create
```

A.11.4 New Directory Site Environment for Database Copy Procedure

New directory site's environment for database copy procedure is explained with an example.

In the example shown throughout this chapter, the new directory site's environment is as follows:

```
Hostname = dsm-sun
Domain name = example.com
ORACLE_BASE = /private1/oracle/app/oracle
ORACLE_HOME = /private1/oracle/app/oracle/product/OraHome_1
ORACLE_SID  = NLDAP
LD_LIBRARY_PATH = $ORACLE_HOME/lib
NLS_LANG = AMERICAN_AMERICA.AL32UTF8
datafile location = /private1/oracle/oradata/NLDAP
Dump destination =  /private1/oracle/app/oracle/admin/NLDAP/pfile,
                    /private1/oracle/app/oracle/admin/NLDAP/bdump,
                    /private1/oracle/app/oracle/admin/NLDAP/cdump,
                    /private1/oracle/app/oracle/admin/NLDAP/udump,
                    /private1/oracle/app/oracle/admin/NLDAP/create
```

Everything except the hostname and domain name is created during installation of the Oracle Database.

A.11.5 Adding a Directory Node by using Database Copy Procedure

You can add a directory node by using the database copy procedure.

This section contains the following topics:

- Setting Up Sponsor Node
- Setting Up New Node
- Running LDAP-Based Replication

A.11.5.1 Setting Up Sponsor Node

Perform the following activities to set up the sponsor node:

- Installing Identity Management with Oracle Internet Directory on Sponsor Node
- Checking the Status of Oracle Internet Directory
- Stopping Oracle Internet Directory
- Shutting Down Database and Oracle Net Services Listener on Sponsor Node
- Renaming Trace File to newdb.sql
- Editing newdb.sql File on the Sponsor Node
- Creating and Copying Initialization Parameter File
- Editing the Initialization Parameter File on the Sponsor Node
- Including Connection Details for the New Node in tnsnames.ora File
- Creating a Compressed Archive File of All Data Files

A.11.5.1.1 Installing Identity Management with Oracle Internet Directory on Sponsor Node

Install Identity Management with the Oracle Internet Directory component on the sponsor node. See Installing and Configuring Oracle Identity Management in *Installing and Configuring Oracle Internet Directory*.

> **See Also:**
>
> About the Directories for Installation and Configuration in *Planning an Installation of Oracle Fusion Middleware*.

A.11.5.1.2 Checking the Status of Oracle Internet Directory

To check the status of Oracle Internet Directory, type:

```
oid_instanceStatus(instanceName = 'instance-name')
```

The status should be `Alive`.

A.11.5.1.3 Stopping Oracle Internet Directory

Stop Oracle Internet Directory using the following `WLST` command:

```
shutdown(name='instance-name',type='OID')
```

A.11.5.1.4 Shutting Down Database and Oracle Net Services Listener on Sponsor Node

Shut down the database and Oracle Net Services listener on the sponsor node. By default, the listener name is LISTENER. Type:

```
$ lsnrctl stop
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> shutdown
SQL> exit
```

A.11.5.1.5 Renaming Trace File to newdb.sql

Rename the trace file created earlier to `newdb.sql`, under the same directory.

```
$ cp ORACLE_SID_ora_processid.trc newdb.sql
```

A.11.5.1.6 Editing newdb.sql File on the Sponsor Node

Edit the newdb.sql file on the server node as mentioned below:

1. On the sponsor node, open `newdb.sql` in a text editor and delete all the lines except the `STARTUP NOMOUNT` and `CREATE CONTROLFILE` statements. After deleting those lines, `newdb.sql` should look like this:

   ```
   STARTUP NOMOUNT
   CREATE CONTROLFILE REUSE SET DATABASE "LDAP" RESETLOGS  NOARCHIVELOG
       MAXLOGFILES 16
       MAXLOGMEMBERS 3
   ```

```
    MAXDATAFILES 100    MAXINSTANCES 8
    MAXLOGHISTORY 454
LOGFILE
  GROUP 1 '/private/oracle/oradata/LDAP/redo01.log'  SIZE 10M,
  GROUP 2 '/private/oracle/oradata/LDAP/redo02.log'  SIZE 10M,
  GROUP 3 '/private/oracle/oradata/LDAP/redo03.log'  SIZE 10M
-- STANDBY LOGFILE
DATAFILE
  '/private/oracle/oradata/LDAP/system01.dbf',
  '/private/oracle/oradata/LDAP/sysaux01.dbf',
  '/private/oracle/oradata/LDAP/users01.dbf',
  '/private/oracle/oradata/LDAP/dcm.dbf',
  '/private/oracle/oradata/LDAP/portal.dbf',
  '/private/oracle/oradata/LDAP/ptldoc.dbf',
  '/private/oracle/oradata/LDAP/ptlidx.dbf',
  '/private/oracle/oradata/LDAP/ptllog.dbf',
  '/private/oracle/oradata/LDAP/oca.dbf',
  '/private/oracle/oradata/LDAP/discopltc1.dbf',
  '/private/oracle/oradata/LDAP/discopltm1.dbf',
  '/private/oracle/oradata/LDAP/oss_sys01.dbf',
  '/private/oracle/oradata/LDAP/wcrsys01.dbf',
  '/private/oracle/oradata/LDAP/uddisys01.dbf',
  '/private/oracle/oradata/LDAP/b2b_dt.dbf',
  '/private/oracle/oradata/LDAP/b2b_rt.dbf',
  '/private/oracle/oradata/LDAP/b2b_idx.dbf',
  '/private/oracle/oradata/LDAP/b2b_lob.dbf',
  '/private/oracle/oradata/LDAP/bam.dbf',
  '/private/oracle/oradata/LDAP/orabpel.dbf',
  '/private/oracle/oradata/LDAP/attrs1_oid.dbf',
  '/private/oracle/oradata/LDAP/battrs1_oid.dbf',
  '/private/oracle/oradata/LDAP/gcats1_oid.dbf',
  '/private/oracle/oradata/LDAP/gdefault1_oid.dbf',
  '/private/oracle/oradata/LDAP/svrmg1_oid.dbf',
  '/private/oracle/oradata/LDAP/ias_meta01.dbf',
  '/private/oracle/oradata/LDAP/undotbs.dbf'
CHARACTER SET AL32UTF8
;
```

2. Continue editing the file `newdb.sql` on the sponsor node, as follows:

   a. Change the line:

      ```
      CREATE CONTROLFILE REUSE DATABASE "LDAP" RESETLOGS  NOARCHIVELOG
      ```

      to:

      ```
      CREATE CONTROLFILE REUSE SET DATABASE "NLDAP" RESETLOGS  NOARCHIVELOG
      ```

   b. Modify the UNIX directory location of the database and logfiles to point to the new node site's directory.

   In our example, after these modifications, `newdb.sql` should look like this:

```
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE SET DATABASE "NLDAP" RESETLOGS  NOARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 454
LOGFILE
  GROUP 1 '/private1/oracle/oradata/NLDAP/redo01.log'  SIZE 10M,
  GROUP 2 '/private1/oracle/oradata/NLDAP/redo02.log'  SIZE 10M,
  GROUP 3 '/private1/oracle/oradata/NLDAP/redo03.log'  SIZE 10M
```

```
-- STANDBY LOGFILE
DATAFILE
  '/private1/oracle/oradata/NLDAP/system01.dbf',
  '/private1/oracle/oradata/NLDAP/sysaux01.dbf',
  '/private1/oracle/oradata/NLDAP/users01.dbf',
  '/private1/oracle/oradata/NLDAP/dcm.dbf',
  '/private1/oracle/oradata/NLDAP/portal.dbf',
  '/private1/oracle/oradata/NLDAP/ptldoc.dbf',
  '/private1/oracle/oradata/NLDAP/ptlidx.dbf',
  '/private1/oracle/oradata/NLDAP/ptllog.dbf',
  '/private1/oracle/oradata/NLDAP/oca.dbf',
  '/private1/oracle/oradata/NLDAP/discopltc1.dbf',
  '/private1/oracle/oradata/NLDAP/discopltm1.dbf',
  '/private1/oracle/oradata/NLDAP/oss_sys01.dbf',
  '/private1/oracle/oradata/NLDAP/wcrsys01.dbf',
  '/private1/oracle/oradata/NLDAP/uddisys01.dbf',
  '/private1/oracle/oradata/NLDAP/b2b_dt.dbf',
  '/private1/oracle/oradata/NLDAP/b2b_rt.dbf',
  '/private1/oracle/oradata/NLDAP/b2b_idx.dbf',
  '/private1/oracle/oradata/NLDAP/b2b_lob.dbf',
  '/private1/oracle/oradata/NLDAP/bam.dbf',
  '/private1/oracle/oradata/NLDAP/orabpel.dbf',
  '/private1/oracle/oradata/NLDAP/attrs1_oid.dbf',
  '/private1/oracle/oradata/NLDAP/battrs1_oid.dbf',
  '/private1/oracle/oradata/NLDAP/gcats1_oid.dbf',
  '/private1/oracle/oradata/NLDAP/gdefault1_oid.dbf',
  '/private1/oracle/oradata/NLDAP/svrmg1_oid.dbf',
  '/private1/oracle/oradata/NLDAP/ias_meta01.dbf',
  '/private1/oracle/oradata/NLDAP/undotbs.dbf'
CHARACTER SET AL32UTF8
;
```

A.11.5.1.7 Creating and Copying Initialization Parameter File

Copy the initialization parameter file `init$ORACLE_SID.ora` from the sponsor directory's database to `init$ORACLE_SID_NEW_DIR_DB.ora`. The default location of the initialization parameter file is `$ORACLE_HOME/dbs` on UNIX or Linux and `%ORACLE_HOME%\database` on Windows. In our example, copy `/private/oracle/app/oracle/product/OraHome_1/dbs/initLDAP.ora` to `/private/oracle/app/oracle/product/OraHome_1/dbs/initNLDAP.ora` as shown here:

```
$ cd ORACLE_HOME/dbs
$ cp initLDAP.ora initNLDAP.ora
```

If you are using the server parameter file `spfile$ORACLE_SID.ora` or `spfile.ora` instead of an initialization parameter file, create an initialization parameter file from the server parameter file. For example, if `spfile$ORACLE_SID.ora` is located in the default location `$ORACLE_HOME/dbs` you would type:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> create pfile from spfile
SQL> shutdown immediate
```

This sequence of commands creates an `initLDAP.ora` file at `/private/oracle/app/oracle/product/OraHome_1` from `spfileLDAP.ora`. If the server parameter file is not located in the default location, you must include the complete path, as shown in the following example:

```
SQL> connect / as sysdba
SQL> create pfile='/private/oracle/initLDAP.ora' from
```

```
 spfile=/private/oracle/initLDAP.ora
SQL> shutdown immediate
```

After you create the initialization file parameter file, create a copy of it as explained at the beginning of this step.

A.11.5.1.8 Editing the Initialization Parameter File on the Sponsor Node

In the new initialization parameter file on the sponsor, make the following changes:

1. Comment out the parameter `JOB_QUEUE_PROCESSES`, if it exists.
2. Change the parameter dbname from `LDAP` to `NLDAP`.
3. If the new site's domain name is different from the sponsor directory's domain name, alter the parameter `db_domain` also.
4. Alter the location of the following parameters to point to the location of the new site.

   ```
   background_dump_dest
   core_dump_dest
   user_dump_dest
   control_files
   db_recovery_file_dest
   ```

5. In addition to these parameters, if your initialization parameter file has any parameters that are node specific, such as `DB_RECOVERY_FILE_DEST` and `DB_CREATE_FILE_DEST`, alter those parameters as well.

In our example, the initialization parameter file `initNLDAP.ora` looks like this after these modifications:

```
*.aq_tm_processes=1
*.background_dump_dest='/private1/oracle/app/oracle/admin/NLDAP/bdump'
*.compatible='10.1.0.2.0'
*.control_files='/private1/oracle/app/oracle/admin/NLDAP/control01.ctl',
       '/private1/oracle/app/oracle/admin/NLDAP/control02.ctl',
'/private1/oracle/app/oracle/admin/NLDAP/control03.ctl'
*.core_dump_dest='/private1/oracle/app/oracle/admin/NLDAP/cdump'
*.db_block_size=8192*.db_cache_size=50331648
*.db_domain='example.com'
*.db_file_multiblock_read_count=16
*.db_name='NLDAP'*.db_recovery_file_dest='/private/oracle1/app/oracle/
flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.dispatchers='(PROTOCOL=TCP)(PRE=oracle.aurora.server.GiopServer)',
              '(PROTOCOL=TCP)(PRE=oracle.aurora.server.SGiopServer)'
*.java_pool_size=67108864#*.job_queue_processes=5
*.large_pool_size=8388608
*.max_commit_propagation_delay=0
*.open_cursors=300
*.pga_aggregate_target=33554432*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.sessions=400
*.shared_pool_size=150994944
*.undo_management='AUTO'
*.undo_tablespace='UNDOTBS'
*.user_dump_dest='/private1/oracle/app/oracle/admin/NLDAP/udump'
```

A.11.5.1.9 Including Connection Details for the New Node in tnsnames.ora File

Edit the `tnsnames.ora` file to include connection details for the new node. Refer to the following sample file:

```
LDAP.ACME.COM =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = rst-sun)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ldap.acme.com)
    )
  )
NLDAP.ACME.COM =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = dsm-sun)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = nldap.acme.com)    )
  )
```

A.11.5.1.10 Creating a Compressed Archive File of All Data Files

Create an archive of all the data files and compress the archived file. Be sure to include all the files listed under DATAFILE in `newdb.sql`.

As an example, you could use the following commands to go to the database file location and generate a compressed archive called `OID_DB.tar.Z`:

```
$ cd $ORACLE_BASE/oradata/$ORACLE_SID
$ tar -cvf OID_db.tar *.dbf
$ compress OID_db.tar
```

A.11.5.2 Setting Up New Node

Perform the following activities to set up the new node:

- Installing Oracle Database on New Node
- Creating Data File, Dump, and Trace File Directories in New Node
- Copying Archived File from Sponsor Node to New Node
- Copying Initialization Parameter File from Sponsor Node to New Node
- Ensuring Non-existence of spfileNLDAP.ora and spfile.ora Files
- Copying newdb.sql File From Sponsor Node To New Node
- Setting Environment Variables on New Node
- Running newdb.sql on the New Node by Using SQL*Plus
- Starting the Database and Listener on the New Node
- Changing the Global Database Name on the New Node
- Adding Temporary File to the Tablespace on the New Node
- Configuring Oracle Internet Directory on the New Node
- Stopping Oracle Internet Directory
- Deleting Wallet Files and Resetting ODS Password at the New Node
- Resetting Oracle Internet Directory Password on the New Node
- Starting Oracle Internet Directory Processes on the New Node
- Resetting Replica ID of the New Node
- Recreating Relative Replica Entries for New Node

- Changing Attributes of Replica Subentry
- Stopping Oracle Internet Directory Processes
- Cleaning Up Change Log Tables at New Node

A.11.5.2.1 Installing Oracle Database on New Node

Install Oracle Database on the new node using the software only option. See the *Oracle Database Installation Guide* for your platform and *Installing and Configuring Oracle Internet Directory*.

A.11.5.2.2 Creating Data File, Dump, and Trace File Directories in New Node

When the software-only install on the new node has completed, the following directory exists:

```
/private1/oracle/app/oracle/product/OraHome_1/diag/rdbms
```

Create the following directories on the new node:

- Datafile location: `/private1/oracle/app/oracle/oradata/NLDAP`
- Dump destinations:

  ```
  /private1/oracle/app/oracle/admin/NLDAP/adump
  ```

  ```
  /private1/oracle/app/oracle/admin/NLDAP/udump
  ```

  ```
  /private1/oracle/app/oracle/flash_recovery_area
  ```

- Trace file location: `/private1/oracle/app/oracle/product/OraHome_1/diag/rdbms/nldap/nldap/trace`

A.11.5.2.3 Copying Archived File from Sponsor Node to New Node

Copy the archived file created on the sponsor node to the new node, using FTP or another appropriate tool. Change directory to the database file location on the new node, then use FTP to copy the archived file from rst-sun.

```
$ cd /private1/oracle/app/oracle/oradata/NLDAP
$ ftp
ftp> open rst-sun
Connected to rst-sun.us.example.com.
220 rst-sun FTP server (UNIX(r) System V Release 4.0) ready.
Name (rst-sun:oracle):
331 Password required for oracle.
Password:
230 User oracle logged in.
ftp> cd /private/oracle/oradata/LDAP
250 CWD command successful.
ftp> binary
200 Type set to I.
ftp> mget oradb.tar.Z
```

If the data files are very large (several gigabytes or terabytes) and the network bandwidth is low, consider using media, such as tape or disk, to move the compressed file from the sponsor to the new node.

Extract the archived file on the new node. For example:

```
$ uncompress oradb.tar.Z
$ tar xvf oradb.tar
```

Ensure that the data files are extracted to the correct directory. In our example, it is /private1/oracle/oradata/NLDAP

A.11.5.2.4 Copying Initialization Parameter File from Sponsor Node to New Node

Copy the initialization parameter file `initLDAP.ora` from the sponsor node (rst-sun) to the new node under the UNIX directory `$ORACLE_HOME/dbs` using FTP or another appropriate tool. Ensure that the contents of the copied file `initLDAP.ora` are valid after copying.

In addition, also copy the database password file `orclpw`*`ORACLE_SID`* from the sponsor node to the new node.

A.11.5.2.5 Ensuring Non-existence of spfileNLDAP.ora and spfile.ora Files

On the new node, ensure that the following files do not exist in the directory `$ORACLE_HOME/dbs` on UNIX or `ORACLE_HOME\database` in Windows:

- `spfileNLDAP.ora`
- `spfile.ora`

If either of these files exists, the Oracle database uses that file instead of the `initNLDAP.ora` file you copied from the sponsor node.

A.11.5.2.6 Copying newdb.sql File From Sponsor Node To New Node

Using FTP or another appropriate tool, copy the file `newdb.sql` you created on the sponsor node in Step 2 to the new node. For example:

```
$ cd /private1/oracle/app/oracle/admin/NLDAP/udump
$ ftp
ftp> open rst-sun
ftp> cd /private1/oracle/app/oracle/admin/LDAP/udump
ftp> mget newdb.sql
```

A.11.5.2.7 Setting Environment Variables on New Node

At the UNIX shell prompt on the new node, set ORACLE_BASE, ORACLE_HOME and ORACLE_SID environment variables. For example (using the C shell):

```
$ setenv ORACLE_BASE /private1/oracle/app/oracle
$ setenv ORACLE_HOME /private1/oracle/app/oracle/product/OraHome_1
$ setenv ORACLE_SID NLDAP
```

A.11.5.2.8 Running newdb.sql on the New Node by Using SQL*Plus

In the same UNIX shell, execute `newdb.sql` using SQL*Plus as shown in the following example:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> @newdb.sql
SQL> shutdown normal
SQL>exit
```

A.11.5.2.9 Starting the Database and Listener on the New Node

Start up the database and listener as follows:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> startup mount
SQL> alter database open resetlogs
SQL> exit
$ lsnrctl start
```

A.11.5.2.10 Changing the Global Database Name on the New Node

Change the global database name of the new node.

```
SQL> connect / as sysdba
SQL> alter database rename global_name to NLDAP;
SQL> exit
```

Note that the new node ORACLE_SID IS NLDAP.

A.11.5.2.11 Adding Temporary File to the Tablespace on the New Node

Add a temporary file to the tablespace using the following command:

```
SQL> connect / as sysdba
SQL> ALTER TABLESPACE TEMP ADD TEMPFILE 'temp01.dbf' size 2000k;
SQL> exit
```

The value `2000K` is just an example. Determine your own value based on the requirements of your environment.

A.11.5.2.12 Configuring Oracle Internet Directory on the New Node

Configure Oracle Internet Directory on the new node:

1. Install WebLogic Server. See Planning the Oracle WebLogic Server Installation.
2. Install Oracle Internet Directory. See Installing and Configuring Oracle Internet Directory*Installing and Configuring Oracle Internet Directory*.

> **Note:**
>
> When you configure Oracle Internet Directory on the new node, you must specify the information for the new node database.

A.11.5.2.13 Stopping Oracle Internet Directory

Stop Oracle Internet Directory by using `wlst` command, as given below:

```
shutdown(name='instance-name',type='OID')
```

A.11.5.2.14 Deleting Wallet Files and Resetting ODS Password at the New Node

On the new node, delete the wallet files `oidpwdlldap1` and `oidpwdr*` and reset the ODS password.

```
$ cd DOMAIN_HOME/config/fmwconfig/components/OID/admin
$ rm oidpwdlldap1 oidpwdr*
```

A.11.5.2.15 Resetting Oracle Internet Directory Password on the New Node

On the new node, reset the password and start the Oracle Internet Directory processes.

```
oidpasswd connect=oiddb create_wallet=true
```

you are prompted to provide the current database password, enter a new database password, and confirm the new password.

A.11.5.2.16 Starting Oracle Internet Directory Processes on the New Node

Start Oracle Internet Directory on the new node by using the `wlst` command, as given below:

```
start(name='instance-name',type='OID')
```

A.11.5.2.17 Resetting Replica ID of the New Node

At this point, Oracle Internet Directory on the new node is up and running. The replicaid value in new Oracle Internet Directory node, however, still has the replica id of the sponsor's node. Therefore, you must reset new node's replicaid. The new value of replicaid must be of the form *hostname_sid* where:

- *hostname* is the host name of the new node, without the domain name
- *sid* is the `ORACLE_SID` of the new node database

Ensure that all letters of the replicaid are in lower case.

1. Create a file, `chgrid.ldif`, with the following contents:

   ```
   dn:
   changetype: modify
   replace: orclreplicaid
   orclreplicaid: new_replicaid
   ```

2. Using the `ldapmodify` tool, change the replicaid:

   ```
   $ORACLE_HOME/bin/ldapmodify -h new_node_host -p port_ldap_server -D cn=orcladmin -q -
   f chgrid.ldif
   ```

A.11.5.2.18 Recreating Relative Replica Entries for New Node

Because the replica id of the new node was changed, you must re-create the relative replica entries for the new node, as follows:

```
$ remtool -pcleanup -bind "new_node_host:new_node_port/new_node_repl_pswd"
```

The `remtool` command does report an error and prompt for input because there are no replica entries that correspond with the new replica id yet. The `remtool` command uses your responses to rectify the error. Here is an example, with user input shown in boldface:

```
remtool -pcleanup -bind "new_node_host:new_node_port/new_node_repl_pswd"
Error occurred while getting replication configuration information.
This tool will try to rectify the problem if super user DN and password are provided.
Do you want to continue? [y/n] : y
Enter superuser DN                          : cn=orcladmin
Enter superuser password                    :
Enter new password of replication DN     :
Reenter new password of replication DN  :
DRG identified by replica ldap://new_node_host:new_node_port (new_replicaid) will be
cleaned up.
```

```
Do you want to continue? [y/n] : y
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
Replica replica ldap://new_node_host:new_node_port (new_replicaid) has been cleaned up.
```

A.11.5.2.19 Changing Attributes of Replica Subentry

In addition to renaming the replica subentry, you must change the `orclreplicauri`, `orclreplicasecondaryuri` and `orclreplicastate` attributes of the replica subentry. You must modify the `orclreplicauri` and `orclreplicasecondaryuri` attributes to contain the URI of the new node's LDAP server. You must set the `orclreplicastate` attribute must be set to `6`, which specifies to `remtool` that this a database copy-based addnode.

To change the values, proceed as follows.

1. Create an LDIF file, `modsubentry.ldif`, with the following contents:

   ```
   dn: orclreplicaid=new_replicaid,cn=replication configuration
   changetype: modify
   replace: orclreplicauri
   # Use your host name and port number
   # where ldap server is listening
   orclreplicauri: ldap://new_node_host:new_node_port/  ---
   replace:orclreplicasecondaryuri
   # Use your fully qualified host name and
   # the port number where ldap server is listening  orclreplicasecondaryuri:
   ldap://new_node_host_with_domain_name:new_node_port/  ----
   replace:orclreplicastate
   orclreplicastate: 6
   ```

2. Using the `ldapmodify` tool, apply the changes to the directory:

   ```
   $ ldapmodify -h new_node_host -p port_of_ldap_server -f modsubentry.ldif
   ```

A.11.5.2.20 Stopping Oracle Internet Directory Processes

To stop Oracle Internet Directory processes, execute the following command:

```
$shutdown(name='instance-name',type='OID')
```

A.11.5.2.21 Cleaning Up Change Log Tables at New Node

Clean up the changelog tables at the new node.

```
$ sqlplus /nolog
SQL> connect ods/ods_password;
SQL> truncate table ods.ods_chg_log;
SQL> truncate table ods.ods_chg_stat;
SQL> truncate table ods.asr_chg_log;
```

A.11.5.3 Running LDAP-Based Replication

If the new node is a full LDAP-based replication replica, configure LDAP-based Replication and add the full replica as fan-out, as follows:

1. Make sure that the database and Oracle Internet Directory server is running at the sponsor node.

   On the sponsor node, type:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> startup
SQL> exit
$ lsnrctl start
Start OID
$DOMAIN_HOME/bin/startComponent.sh <instance-name>
```

Check the status of Oracle Internet Directory by typing:

```
$oid_instanceStatus(instanceName = 'instance-name')
```

2. Configure LDAP-based replication using `remtool`, as follows:

```
remtool -paddnode
```

3. Initialize the replication change status of the new replication agreement.
   - Get the maximum used change number from the sponsor node:

     ```
     $ ldapsearch -h sponsor_node_host -p sponor_node_port -b " " \
        -s base "objectclass=*" lastchangenumber
     ```

   - Create an LDIF file chgstatus.ldif, with the following contents:

     ```
     dn: orclagreementid=new_agreement_id,orclreplicaid=new_replicaid,cn=replication
     configuration
     changetype: modify
     replace: orcllastappliedchangenumber;transport$sponsor_replicaid$new_replicaid
     orcllastappliedchangenumber;transport$sponsor_replicaid$new_replicaid:
     Number_from_ldapsearch
     -
     replace: orcllastappliedchangenumber;apply$sponsor_replicaid$new_replicaid
     orcllastappliedchangenumber;apply$sponsor_replicaid$new_replicaid:
     Number_from_ldapsearch
     ```

     where *Number_from_ldapsearch* refers to the maximum change number from the sponsor node that you obtained using the `ldapsearch` command.

   - Using `ldapmodify`, apply the change to both the sponsor node and the new node:

     ```
     ldapmodify -p sponsor_node_port -h sponsor_node_host -v \
        -f chgstatus.ldif
     ldapmodify -p new_node_port -h new_node_host -v -f chgstatus.ldif
     ```

4. Start up Oracle Internet Directory and the replication server on all the nodes. For one-way replication, you only need to start the replication server on the consumer node.

A.12 RFCs Supported by Oracle Internet Directory

Table A-27 contains a list of the RFCs currently supported by Oracle Internet Directory. Some of these have been obsoleted or updated by newer RFCs. Oracle Internet Directory is continuously being updated to ensure that it conforms to the newer documents.

Table A-27 Supported RFCs

Number	Title
RFC 1274	The COSINE and Internet X.500 Schema
RFC 1488	The X.500 String Representation of Standard Attribute Syntaxes

Table A-27 (Cont.) Supported RFCs

Number	Title
RFC 1777	Lightweight Directory Access Protocol
RFC 1778	The String Representation of Standard Attribute Syntaxes
RFC 1779	A String Representation of Distinguished Names
RFC 1960	A String Representation of LDAP Search Filters
RFC 2079	Definition of an X.500 Attribute Type and an Object Class to Hold Uniform Resource Identifiers (URIs)
RFC 2247	Using Domains in LDAP/X.500 Distinguished Names
RFC 2251	Lightweight Directory Access Protocol (v3)
RFC 2252	Lightweight Directory Access Protocol (v3) Attribute Syntax Definitions
RFC 2253	Lightweight Directory Access Protocol (v3) UTF-8 String Representation of Distinguished Names
RFC 2254	The String Representation of LDAP Search Filters
RFC 2255	The LDAP URL Format
RFC 2256	A summary of the X500 (96) User Schema for use with LDAPv3
RFC 2307	An Approach for Using LDAP as a Network Information Service
RFC 2377	Naming Plan for Internet Directory-Enabled Applications
RFC 2396	Uniform Resource Identifiers (URI): Generic Syntax
RFC 2596	Use of Language Codes in LDAP
RFC 2696	LDAP Control Extension for Simple Paged Results Manipulation
RFC 2713	Schema for Representing Java(tm) Objects in an LDAP Directory
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 2849	The LDAP Data Interchange Format (LDIF) - Technical Specification
RFC 2891	LDAP Control Extension for Server Side Sorting of Search Results
RFC 3112	LDAP Authentication Password Schema
RFC 3296	Named Subordinate References in Lightweight Directory Access Protocol (LDAP) Directories
RFC 3377	Core LDAP Requirements
RFC 3671	Collective Attributes in the Lightweight Directory Access Protocol (LDAP)
RFC 3673	Search results returning all operational attributes
RFC 4122	Universally Unique IDentifier (UUID) URN Namespace
RFC 4514	String Representation of Distinguished Names
RFC 5805	Lightweight Directory Access Protocol (LDAP) Transactions

Oracle Internet Directory's base schema is continually revised to comply with new RFC revisions.

> **See Also:**
>
> The Internet Engineering Task Force home page at http://www.ietf.org/.

A.13 Managing Oracle Directory Services Manager's Java Key Store

This appendix describes how to manage the Oracle Directory Services Manager (ODSM) Java Key Store (JKS), including retrieving the JKS password, listing the contents of the `odsm.cer` JKS, and deleting expired certificates. It includes the following sections:

- Introduction to Managing ODSM's Java Key Store
- Retrieving ODSM's Java Key Store Password
- Listing the Contents of odsm.cer Java Key Store
- Deleting Expired Certificates

A.13.1 Introduction to Managing ODSM's Java Key Store

Oracle Directory Services Manager (ODSM) stores its private key, certificate and trusted certificates in a Java Key Store (JKS).

As administrator, you are responsible for managing ODSM's JKS. One important task you must perform is to remove ODSM certificates from the JKS when they have expired. This appendix explains how.

The first time ODSM is invoked, it generates a random password and assigns the password to its JKS. The JKS file has the name `odsm.cer`. The file resides in a directory with a name of the form:

$DOMAIN_HOME`/config/fmwconfig/servers/AdminServer/applications/odsm/conf`

ODSM stores the password to its JDK in the Credential Store Framework (CSF), a secure storage framework provided by Oracle. The WebLogic server administrator can retrieve the JDK password stored in the CSF.

ODSM also generates a self-signed certificate for itself and stores it in its JKS. This self-signed certificate is valid for 15000 days from the date of generation. This self-signed certificate is intended for testing purposes only. Oracle recommends replacing this self-signed certificate with a certificate signed by a Certificate Authority (CA) for production purposes.

There is no web-based tool for managing a JKS. To manage ODSM's JKS, you use `keytool`, a command-line tool shipped with the Oracle JRE or JDK.

See Also:

- The section about configuring the credential store in the *Oracle Fusion Middleware Application Security Guide* for more information about the CSF.
- For information about Oracle Java, including the JavaTM Cryptography Architecture API Specification & Reference and keytool - Key and Certificate Management Tool, see:

  `http://www.oracle.com/technetwork/java/index.html`

A.13.2 Retrieving ODSM's Java Key Store Password

To manage Oracle Directory Services Manager's Java Key Store, you must first retrieve Oracle Directory Services Manager's Java Key Store password.

There are two methods for retrieving this password:

- Retreiving Password Using Enterprise Manager Fusion Middleware Control
- Retrieving Password Using a Python Script

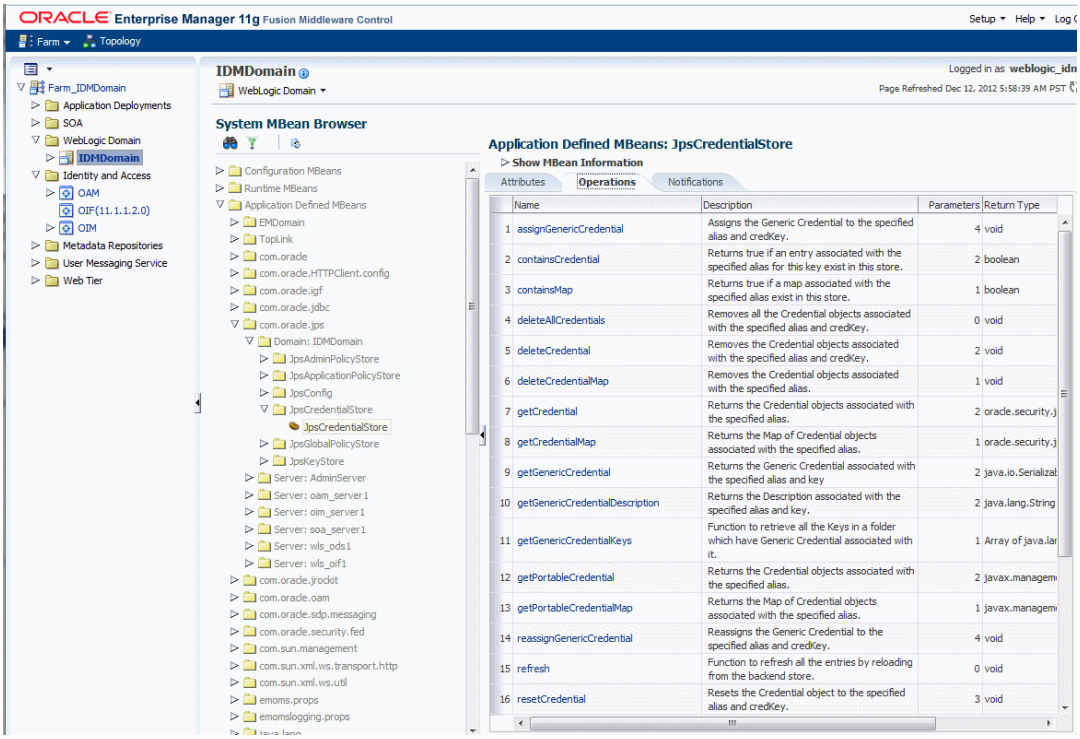
A.13.2.1 Retreiving Password Using Enterprise Manager Fusion Middleware Control

Use Enterprise Manager to retrieve the Oracle Directory Services Manager's Java Key Store password.

Perform the following steps:

1. Connect to Enterprise Manager as `weblogic administrator`.
2. On left side of the Enterprise Manager navigation panel, expand **WebLogic Domain** and select the domain in which Oracle Directory Services Manager is deployed.
3. On right side of the Enterprise Manager navigation panel, click the **Weblogic Domain** menu and select **System MBean Browser**.
4. In the System MBean browser, expand **Application Defined MBeans** > **com.oracle.jps** > **Domain:** *NameOfTheDomainWhereODSMisDeployed* > **JPSCredentialStore** > **JPSCredential Store**.
5. Click the Operations tab shown on the details pane.
6. Click the **getPortableCredential** operation as shown in the following figure.

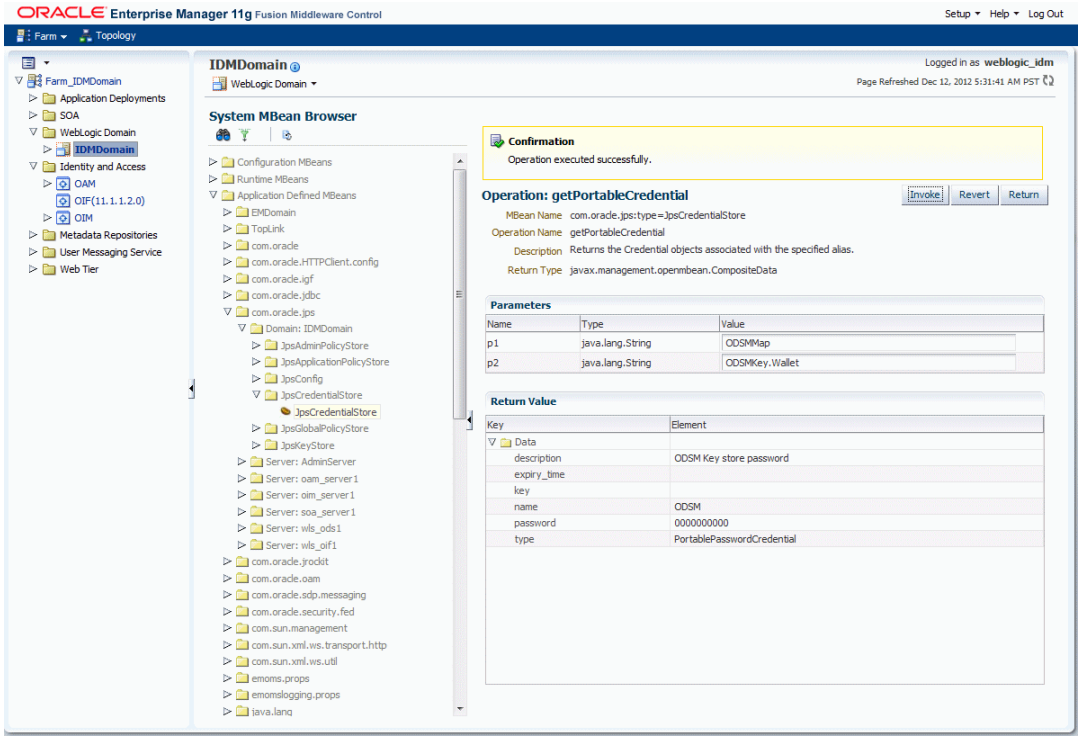
Figure A-4 getPortableCredential Operation



7. When the next page displays, you must enter parameters for the **getPortableCredential Method**.
   - For **P1**, enter `ODSMMap`.
   - For **P2**, enter `ODSMKey.Wallet`.
8. Click the **Invoke** button.

   The Return Value table is displayed below the Parameters table, as shown in the following figure. The Oracle Directory Services Manager Java Key Store password is displayed in the **password** field.

Figure A-5 getPortableCredential Operation



A.13.2.2 Retrieving Password Using a Python Script

If you do not have Enterprise Manager, you can retrieve the Oracle Directory Services Manager's Java Key Store password by using a Python script.

To retrieve this password, perform the following steps:

1. Create a `.py` file (for example, `odsm.py`) with the following contents:

```
import sys,getopt
from oracle.security.jps.mas.mgmt.jmx.credstore import PortableCredential
connect(sys.argv[1], sys.argv[2], sys.argv[3])
domainRuntime()
params= ["ODSMMap", "ODSMKey.Wallet"]
sign=["java.lang.String", "java.lang.String"]
on=ObjectName("com.oracle.jps:type=JpsCredentialStore")
cred = None
cred = mbs.invoke(on, "getPortableCredential", params, sign)
if cred != None:
   credObject = PortableCredential.from(cred)
   print credObject
   print "ODSM Java Key Store Password: " + String.valueOf(credObject.getPassword())
```

2. Execute the following command:

```
$MW_HOME/oracle_common/common/bin/wlst.sh odsm.py <wls_admin_user>
<wls_admin_password> t3://<adminserver_host>:<adminserver_port>
```

For example,

```
$MW_HOME/oracle_common/common/bin/wlst.sh odsm.py weblogic password
t3://myadminserver:7001
```

Running this script changes the output location to the `domainRuntime` tree, which is a read-only tree with `DomainMBean` as the root.

> **Note:**
>
> For help, type **`help(domainRuntime)`** at the command line.

```
[Name : ODSM, Description : ODSM Key store password, expiry Date : null]
ODSM Java Key Store Password: XXXXXXXXXX
```

> **See Also:**
>
> For more information, refer to the following publications:
>
> - Managing Keys and Certificates with the Keystore Service in *Oracle Fusion Middleware Securing Applications with Oracle Platform Security Services*

A.13.3 Listing the Contents of odsm.cer Java Key Store

After you retrieve the JKS password, you can manage the JKS by using `keytool`.

To list the contents of `odsm.cer`, use the `keytool` command, as follows:

```
cd directory_where_odsm.cer_resides
JAVA_HOME/bin/keytool -list -keystore odsm.cer \
   -storepass password_obtained_from_CSF
```

For example:

```
$ cd DOMAIN_HOME/config/fmwconfig/servers/AdminServer/applications/odsm/conf
$ JAVA_HOME/bin/keytool -list -keystore odsm.cer -storepass "&M)S86)/RB" -v


Keystore type: JKS
Keystore provider: SUN

Your keystore contains 2 entries

Alias name: serverselfsigned
Creation date: Dec 26, 2008
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=OVD, OU=Development, O=Oracle, L=Redwood Shores, ST=California, C=US
Issuer: CN=OVD, OU=Development, O=Oracle, L=Redwood Shores, ST=California, C=US
Serial number: 495586b6
Valid from: Fri Dec 26 17:36:54 PST 2008 until: Wed Jun 24 18:36:54 PDT 2009
Certificate fingerprints:
         MD5:  6C:11:16:F3:88:8D:18:67:35:1E:16:5B:3E:03:8A:93
         SHA1: F4:91:39:AE:8B:AC:46:B8:5D:CB:D9:A4:65:BE:D2:75:08:17:DF:D0
         Signature algorithm name: SHA1withRSA         Version: 3


*******************************************
*******************************************
```

```
Alias name: cn=rootca, o=oracle, c=us (0)
Creation date: Dec 31, 2008
Entry type: trustedCertEntry

Owner: CN=RootCA, O=Oracle, C=US
Issuer: CN=RootCA, O=Oracle, C=US
Serial number: 0
Valid from: Tue Dec 30 02:33:11 PST 2008 until: Mon Jan 24 02:33:11 PST 2050
Certificate fingerprints:
         MD5:  72:31:7B:24:C9:72:E3:90:37:38:68:40:79:D1:0B:4B
         SHA1: D2:17:84:1E:19:23:02:05:61:42:A9:F4:16:C8:93:84:E8:20:02:FF
         Signature algorithm name: MD5withRSA
         Version: 1


*******************************************
*******************************************
```

A.13.4 Deleting Expired Certificates

There is no automatic mechanism for removing certificates from the JDK when they expire.

As administrator, you must determine when a certificate has expired and remove it.

This section contains the following topics:

- Determining the Expiration Date of a Certificate
- Deleting a Certificate

A.13.4.1 Determining the Expiration Date of a Certificate

This section helps you to determine the expiration date of a certificate.

As explained in Listing the Contents of odsm.cer Java Key Store, you list all certificates in `odsm.cer` by using `keytool`. The listing contains the valid dates for each certificate. For example, the following certificate is valid until Sat Oct 31 09:41:23 PDT 2008:

```
Alias name: cn=ovd, ou=development, o=MyCompany, l=redwood shores,
st=california, c=us (1241455283)
Creation date: May 5, 2008
Entry type: trustedCertEntry

Owner: CN=OVD, OU=Development, O=MyCompany, L=Redwood Shores, ST=California, C=US
Issuer: CN=OVD, OU=Development, O=Oracle, L=Redwood Shores, ST=California, C=US
Serial number: 49ff1ab3
Valid from: Mon May 04 09:41:23 PDT 2008 until: Sat Oct 31 09:41:23 PDT 2008
Certificate fingerprints:
MD5: 93:0E:41:5E:95:88:71:BD:8A:49:ED:A9:29:3B:0A:1E
SHA1: 84:C6:75:60:D9:BE:7B:CA:D6:8B:B5:4B:97:E4:20:39:44:82:FE:93
Signature algorithm name: SHA1withRSA
Version: 3
```

If certificate's validity period has expired, delete it using `keytool` as explained in the next section.

A.13.4.2 Deleting a Certificate

To delete a certificate in `odsm.cer`, use `keytool`.

Execute the following command:

```
cd directory_where_odsm.cer_is_present
JAVA_HOME/bin/keytool -delete -keystore odsm.cer
-storepass password_obtained_from_CSF -alias "cn=rootca, o=oracle, c=us (0)"
```

For example

```
$> JAVA_HOME/bin/keytool -delete -keystore odsm.cer \
   -storepass  "&M)S86)/RB"  -alias "cn=rootca, o=oracle, c=us (0)"
[Storing odsm.cer]
```

A.14 Starting and Stopping the Oracle Stack

This appendix describes how to start and stop the components of the Oracle stack in a specific order. It includes the following sections:

> **Note:**
>
> For an Oracle Autonomous Transaction Processing-Shared (ATP-S) database, before starting/stopping the Oracle WebLogic Administration Server, OID system components, or the Node Manager, you must set the `TNS_ADMIN` property to `<$DOMAIN_HOME>/config/fmwconfig/components/OID/config/` using the following command:
>
> ```
> export TNS_ADMIN=<$DOMAIN_HOME>/config/fmwconfig/components/OID/config
> ```
>
> To update the contents of the directory pointed to by TNS_ADMIN, see Prerequisites for ATP-S.

- Starting the Stack
- Stopping the Stack

A.14.1 Starting the Stack

This section describes the procedure to start the stack.

Start the stack components in the following order.

1. Start the Oracle Database.
   a. In the Database `ORACLE_HOME`, set the `ORACLE_SID`, `ORACLE_HOME` and `PATH` environment variables to the appropriate values.
   b. Start the listener.

      ```
      ORACLE_HOME/bin/lsnrctl start
      ```

   c. Start the database.

```
ORACLE_HOME/bin/sqlplus "/as sysdba"
startup
```

2. Start the Oracle WebLogic Administration Server.

> **Note:**
>
> If you start the Oracle WebLogic Administration Server from the command line as shown, it runs in the foreground and prints output to the screen. You can, however, run the server in the background by using `nohup` at the beginning of the command line. This sends all output to the file `nohup.out` and prevents the script from prompting you for *`USER_NAME`* and *`PASSWORD`*. To pass parameters to `StartWebLogic.sh` when using `nohup`, you can use a boot identity file. See Starting and Stopping Servers in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

```
(UNIX) DOMAIN_HOME/bin/startWebLogic.sh
(Windows) DOMAIN_HOME\bin\startWebLogic.cmd
```

When executing these scripts:

- The default value for *`DOMAIN_NAME`* is `base_domain`
- You will be prompted for values for *`USER_NAME`* and *`PASSWORD`* if you do not provide them as options when you execute the script.

3. Ensure that the Node Manager is running. If, for some reason, the Node Manager is not running, start it.

```
(UNIX) DOMAIN_HOME/bin/startNodeManager.sh
(Windows) DOMAIN_HOME\bin\startNodeManager.cmd
```

4. Start WebLogic system components, such as Oracle Internet Directory.

```
(UNIX) DOMAIN_HOME/bin/startComponent.sh component_name
(Windows) DOMAIN_HOME\bin\startComponent.cmd component_name
```

You can verify that the system components have started in a WLST shell by executing:

```
oid_instanceStatus(instanceName = 'instance-name')
```

5. Start WebLogic Managed Servers, such as Oracle Directory Integration Platform.

```
(UNIX) DOMAIN_HOME/bin/startManagedWebLogic.sh managed_server_name admin_url
(Windows) DOMAIN_HOME\bin\startManagedWebLogic.cmd managed_server_name admin_url
```

When prompted, enter your user name and password.

To start managed components in the background, you can use the Oracle WebLogic Administration Console. See Overview of Administration Consoles in *Understanding Oracle WebLogic Server*.

You can view the status of WebLogic managed components with Oracle Enterprise Manager Fusion Middleware Control. See Overview of Using Fusion Middleware Control to Manage Oracle Internet Directory .

A.14.2 Stopping the Stack

You can stop the Administration Server and all the managed servers by using Oracle WebLogic Administration Console.

See Overview of Administration Consoles in *Understanding Oracle WebLogic Server*.

To stop the stack components from the command line, issue the commands in the following order.

1. Stop WebLogic Managed Servers such as `WLS_ODS1`. The Oracle Directory Integration Platform (DIP) typically runs from the `WLS_ODS1` Managed Server.

   ```
   (UNIX) DOMAIN_HOME/bin/stopManagedWebLogic.sh managed_server_name admin_url
   (Windows) DOMAIN_HOME\bin\stopManagedWebLogic.cmd managed_server_name admin_url
   ```

   When prompted, enter your user name and password.

2. Stop WebLogic system components, such as Oracle Internet Directory.

   ```
   (UNIX) DOMAIN_HOME/bin/stopComponent.sh component_name
   (Windows) DOMAIN_HOME\bin\stopComponent.cmd component_name
   ```

3. Stop the WebLogic Administration Server.

   ```
   (UNIX) DOMAIN_HOME/bin/stopWebLogic.sh
   (Windows) DOMAIN_HOME\bin\stopWebLogic.cmd
   ```

4. If you want to stop the Node Manager:

   ```
   (UNIX) DOMAIN_HOME/bin/stopNodeManager.sh
   (Windows) DOMAIN_HOME\bin\stopNodeManager.cmd
   ```

5. Stop the Oracle Database.

   a. In the Database `ORACLE_HOME`, set the `ORACLE_SID`, `ORACLE_HOME`, and `PATH` environment variables to the appropriate values.

   b. Stop the database.

      ```
      ORACLE_HOME/bin/sqlplus "/as sysdba"
      shutdown immediate
      ```

   c. Stop the listener.

      ```
      ORACLE_HOME/bin/lsnrctl stop
      ```

A.15 Performing a Rolling Upgrade

This appendix describes how to perform a rolling upgrade for Oracle Internet Directory if a directory replication group (DRG) is configured for your environment. It includes the following sections:

- About Rolling Upgrade
- Prerequisites for a Rolling Upgrade
- Performing a Rolling Upgrade
- Completing Post-Upgrade Task
- Rolling Upgrade Example

A.15.1 About Rolling Upgrade

To upgrade the Oracle home directories in a DRG, you must shut down all replication servers in the DRG and then upgrade each Oracle home to the latest version individually. If you do not shut down the replication servers, you might encounter issues when upgrading the Oracle home directories.

To avoid the downtime and loss of service associated with this upgrade process, Oracle recommends that you follow a rolling upgrade procedure where each Oracle home directory is upgraded individually, while the other Oracle Internet Directory nodes and the corresponding replication servers in the DRG remain up and running.

A.15.2 Prerequisites for a Rolling Upgrade

Upgrade is only supported for Oracle Internet Directory 11*g* Release 1 (11.1.1.9.0).

A.15.3 Performing a Rolling Upgrade

This section describes the procedure to performing a rolling upgrade.

To upgrade an Oracle Internet Directory DRG:

1. Stop the replication server on the first node you want to upgrade.
2. Suspend the replication from the first node to each of the other nodes:

   ```
   remtool -psuspendrepl -fromnode first_node_host:first_node_port
   ```

   This command lists the replica IDs of all the Oracle Internet Directory nodes along with their respective directory version numbers. You must specify either the replica ID of each of the other nodes as prompted or enter `all`, which suspends replication to all the other nodes.

   See `remtool`, in *Oracle Fusion Middleware Reference for Oracle Identity Management*.
3. Validate that replication has been suspended from the first node to all the other nodes in the DRG by running the following `ldapsearch` command:

   ```
   ldapsearch -h first_node_host -p OID_port  -b "" -s base
   "(objectclass=orclReplAgreementEntry)" orcllastappliedchangenumber
   ```

   Look for `orcllastappliedchangenumber` from the first node to the other nodes with subtype status. It should have value `0` and will be of the form:

   ```
   orcllastappliedchangenumber;status$replica_id_of_first_node$replica_id_of_other_node;
   transport
   ```

   and

   ```
   orcllastappliedchangenumber;apply;status$replica_id_of_first_node$replica_id_of_other
   _node
   ```

   If there are any discrepancies, repeat Step 2.
4. Stop Oracle Internet Directory on the node you are about to upgrade.
5. Patch the node to the current release, as described in the Oracle Fusion Middleware Patching and Upgrade Overview in *Oracle Fusion Middleware Patching Guide*.
6. Start Oracle Internet Directory on the node you have just upgraded.

7. Because no other nodes have been upgraded yet, do not resume replication between the first node and any other node.
8. Stop the replication server on the second node you want to upgrade.
9. Suspend replication from the second node to each of the nodes that have not been upgraded.

   ```
   remtool -psuspendrepl -fromnode second_node_host:second_node_port
   ```

   This command lists the replica IDs of all the Oracle Internet Directory nodes along with their respective directory version numbers. You must specify the replica ID of each of the other nodes that have not been upgraded.
10. Validate that replication has been suspended from the node you are about to upgrade to each node that has not been upgraded, using the same command as in Step 3 against the second node.
11. Stop Oracle Internet Directory on the node you are about to upgrade.
12. Patch the node to the current release, as described in the Oracle Fusion Middleware Patching and Upgrade Overview in *Oracle Fusion Middleware Patching Guide*.
13. Start Oracle Internet Directory on the node you have just upgraded.
14. Now two nodes have been upgraded, so resume replication from the first node you upgraded to the second node you upgraded.

    ```
    remtool -presumerepl -fromnode  first_upgraded_replica_host:port -tonode
    second_upgraded_replica_host:port
    ```
15. Validate that replication has been resumed from first upgraded node to the second upgraded nodes in DRG by running the following `ldapsearch` command:

    ```
    ldapsearch -p OID_port -h first_upgraded_OID_host -b "" -s sub
    "(objectclass=orclReplAgreementEntry)" orcllastappliedchangenumber
    ```

    Look for `orcllastappliedchangenumber` from the first upgraded Oracle Internet Directory node to the second upgraded node with subtype status; it should have a value of 1.

    Repeat this `ldapsearch` command against all other earlier upgraded nodes. If there are any discrepancies, repeat Step 14.
16. Start replication server on the second upgraded replica.
17. Shut down the replication server on the third node you want to upgrade.
18. Suspend replication from the third node to each node that has not yet been upgraded, if any.
19. Validate that replication has been suspended from the third node to each node that has not been upgraded.
20. Stop Oracle Internet Directory and other processes on the third node to be upgraded.
21. Patch the node to the current release, as described in the Oracle Fusion Middleware Patching and Upgrade Overview in *Oracle Fusion Middleware Patching Guide*.
22. Start Oracle Internet Directory on the third node.
23. Resume replication from each node that has already been upgraded to the third node.
24. Validate that replication has been resumed from each upgraded node to the third node.
25. Start replication on the third node

Repeat this procedure for all nodes in the DRG.

Before upgrading a node, suspend replication from that node to each of the nodes that has not yet been upgraded and validate that it is suspended. Stop Oracle Internet Directory on the node you are about to upgrade. Upgrade the node. Start Oracle Internet Directory on the node you just upgraded. Resume replication from nodes that are already upgraded to the node you just upgraded. Start replication on the node you just upgraded.

A.15.4 Completing Post-Upgrade Task

To prevent leaving some nodes in the DRG with inconsistent data after you perform a rolling upgrade, Oracle Internet Directory performs these operations:

To prevent leaving some nodes in the DRG with inconsistent data after you perform a rolling upgrade, Oracle Internet Directory performs these operations:

1. On the Oracle Internet Directory master node that is processing incoming LDAP update requests, a change log entry is generated for each update request. The master node can be any of the nodes in the DRG (node A, B, or C in Rolling Upgrade Example).
2. The changes are subsequently applied to all other nodes in the DRG regardless of whether a node is down or being upgraded when an LDAP request comes in, as long as the nodes in the DRG are brought back up after the rolling upgrade operation completes.

   When a change is applied to a target node, the appropriate conflict resolution logic ensures convergence. The retry logic (by default, 10 times with some cycle delay) handles a transient conflict situation, such as when a target node is down.

A.15.5 Rolling Upgrade Example

The following example includes three Oracle Internet Directory nodes, A, B, and C, in a mulimaster DRG. All nodes need to be upgraded to the current release by using the rolling upgrade procedure

In this example, upgrade is done in Node A first, followed by Node B, and then Node C.

> **Note:**
>
> Before you begin the rolling upgrade procedure, apply any required patches, as described in Prerequisites for a Rolling Upgrade.

To upgrade the nodes:

1. Stop the replication server on Node A.
2. Suspend the replication from Node A to both Node B and Node C. For example:

   ```
   remtool -psuspendrepl  -fromnode  HostA:PortA
   ```

   This command displays the replica IDs of all three Oracle Internet Directory nodes. Provide the replica IDs of Node B and Node C as input to `remtool`. Alternatively, you can enter `all`, which causes `remtool` to automatically select the replica IDs of B and C.
3. Validate that replication has been suspended from Node A to Node B by running the following `ldapsearch` command:

   ```
   ldapsearch -p  PortA  -h  hostA  -b  "" -s  base
   "(objectclass=orclReplAgreementEntry)" orcllastappliedchangenumber
   ```

Look for the `orcllastappliedchangenumber` attribute with subtype `status`. This attribute has the value `0` and is of the form:

```
orcllastappliedchangenumber;apply;status$replica_id_of_node A$replica_id_of_node_B
orcllastappliedchangenumber:status$replicaid_of_node_A$replicaid_of_nodeB;transport
```

If there is a discrepancy in the status value, repeat Step 2.

Similarly, validate that replication has been suspended from Node A to Node C.

4. Stop Oracle Internet Directory and other processes on Node A.
5. Run the Patch Set Assistant (PSA) on Node A to upgrade the node to the current release.
6. Bring up Oracle Internet Directory and the replication server on Node A. Note that there is no node with which Node A should resume replication.
7. Stop the replication server on Node B.
8. Suspend replication from Node B to Node C by executing the following command:

   ```
   remtool -psuspendrepl -fromnode HostB:PortB -tonode HostC:PortC
   ```

9. Validate that replication has been suspended from Node B to Node C by running the following `ldapsearch` command:

   ```
   ldapsearch -p  PortB  -h  hostB  -b  "" -s  base
   "(objectclass=orclReplAgreementEntry)" orcllastappliedchangenumber
   ```

   Look for `orcllastappliedchangenumber` with subtype `status`. It has value `0` and is of the form:

   ```
   orcllastappliedchangenumber;apply;status$replica_id_of_node_B$replica_id_of_node_C
   orcllastappliedchangenumber:status$replicaid_of_node_B$replicaid_of_node_C;transport
   ```

   If there is any discrepancy in the status value, repeat Step 8.
10. Stop Oracle Internet Directory and other processes on Node B.
11. Run the PSA on Node B to upgrade the node to the current release.
12. Bring up Oracle Internet Directory on Node B and resume replication from Node A to Node B, as follows:

    ```
    remtool -presumerepl -fromnode HostA:PortA -tonode HostB:PortB
    ```

13. Validate that replication has been resumed from Node A to Node B by running the following `ldapsearch` command:

    ```
    ldapsearch -p OID_port -h hostA -b "" -s base "(objectclass=
    orclReplAgreementEntry)" orcllastappliedchangenumber
    ```

    Look for `orcllastappliedchangenumber` from the upgraded Node A to Node B with subtype `status`. It should have the value `1`.

    If there is any discrepancy in the status value, repeat Step12.
14. Bring up the replication server on Node B.
15. Shut down the replication server on Node C.
16. Stop Oracle Internet Directory and other processes on Node C. Since there are no other nodes to be upgraded, you do not need to suspend replication.
17. Run PSA on Node C to upgrade the node to the current release.
18. Bring up Oracle Internet Directory on Node C and resume replication from Node A to Node C, as follows.

```
remtool -presumerepl -fromnode HostA:PortA -tonode HostC:PortC
```

Resume replication from Node B to Node C.

```
remtool -presumerepl -fromnode HostB:PortB -tonode HostC:PortC
```

19. Validate that replication has been resumed successfully by performing an `ldapsearch` on Node A and Node B similar to Step 13. Validate replication:
    - From Node A to Node C
    - From Node B to Node C
20. Bring up the replication server on Node C.

A.16 Troubleshooting Oracle Internet Directory

This appendix provides troubleshooting solutions for typical problems and error messages that you might encounter while installing, configuring, or running Oracle Internet Directory. It includes the following sections:

- Problems and Solutions
- Need More Help?

A.16.1 Problems and Solutions

This section describes common Oracle Internet Directory error messages, problems and solutions.

This chapter contains the following topics:

- Installation Errors
- Oracle Database Server Errors
- Directory Server Error Messages and Causes
- Core Dump and Stack Trace Occurs When Oracle Internet Directory Crashes
- TCP/IP Problems
- Troubleshooting Password Policies
- Troubleshooting Directory Performance
- Troubleshooting Port Configuration
- Troubleshooting Starting Oracle Internet Directory
- Oracle Internet Directory Error Due to Interrupted Client Connection
- Troubleshooting Starting, Stopping, and Restarting of the Directory Server
- Troubleshooting Oracle Internet Directory Replication
- Troubleshooting Change Log Garbage Collection
- Troubleshooting Dynamic Password Verifiers
- Troubleshooting Oracle Internet Directory Password Wallets
- Troubleshooting bulkload Errors
- Troubleshooting bulkdelete, bulkmodify, and ldifwrite Errors
- Troubleshooting catalog Errors

- Troubleshooting remtool Errors
- Troubleshooting Server Chaining Error
- View Version Information
- Troubleshooting Oracle Enterprise Manager Fusion Middleware Control and WLST
- Troubleshooting Oracle Directory Services Manager
- Troubleshooting a Locked User Account
- Troubleshooting Policy Store Migration

A.16.1.1 Installation Errors

This section provide information on Oracle database installation errors.

During installation and configuration of the Oracle Database, Oracle recommends that you select the character set AL32UTF8 to avoid possible problems with multibyte characters.

A.16.1.2 Oracle Database Server Errors

Because Oracle Internet Directory relies on Oracle Database, database errors can cause directory server problems.

This section lists some database errors you might see in the Oracle Internet Directory logs.

> **See Also:**
>
> Diagnosing Poor Oracle Database Server Performance.

- Oracle Database Server Connection is Down
- Oracle Database Server Error Due to Interrupted Client Connection
- Oracle Database Server Error Due to Schema Modifications
- Oracle Database TNS Listener Error (ORA-12520)

A.16.1.2.1 Oracle Database Server Connection is Down

Oracle Internet Directory shuts down. You see error ORA-3113 or ORA-3114 in the log file.

Problem

Oracle Internet Directory has lost its connection to Oracle Database.

Solution

Check database and listener status, either directly on the host where they are running, or through. Restart them if necessary. OIDMON automatically detects that the database is up and restarts OIDLDAPD servers.

A.16.1.2.2 Oracle Database Server Error Due to Interrupted Client Connection

You get error `sgslunrRead` or `30SendPort`

Problem

These errors indicates that an LDAP client has disconnected abruptly.

Possible reasons include:

- The client program terminated the connection without performing an unbind or abandon.

  The client machine shut down.
- A network component, such as a load balancer or firewall, broke the connection due to a configured timeout setting.
- The network is down.

Solution

These errors are due to conditions external to the server. If necessary, inform the network administrator.

A.16.1.2.3 Oracle Database Server Error Due to Schema Modifications

You get error `ORA-1562`.

Problem

If you attempt to add more schema components than can fit in the rollback segment space, you encounter this error and the modifications do not commit.

Solution

To solve this, increase the size of the rollback segments in the database server.

A.16.1.2.4 Oracle Database TNS Listener Error (ORA-12520)

Problem

You cannot connect to the Oracle database, or database connectivity is going down and coming up intermittently and is returning the following error when the connectivity goes down:

```
Error ORA-12520: TNS:listener could not find available handler for requested type of server
```

This problem has these potential causes and solutions:

- Troubleshooting Low Oracle Database PROCESSES Parameter Value
- Diagnosing Listener Control Utility (`lsnrctl`) State

A.16.1.2.4.1 Troubleshooting Low Oracle Database PROCESSES Parameter Value

The PROCESSES parameter determines the maximum number of operating system processes that can be connected concurrently to the Oracle Database.

If you see this problem intermittently (that is, the database connectivity is going down and coming up intermittently) and is returning the ORA-12520 error when the connectivity goes down, follow this procedure to troubleshoot the problem:

1. Login to the Oracle Database as the `sysdba` user.
2. Check the value of the PROCESSES parameter. For example:

   ```
   SQL> SHOW PARAMETERS PROCESSES;

   NAME                                 TYPE        VALUE
   ```

```
---------------------------------- ----------- ------------------------------
processes                          integer     150
```

3. Check the `V$RESOURCE_LIMIT` view to display information about global resource use for some of the system resources. For example:

```
SQL> SELECT * FROM V$RESOURCE_LIMIT WHERE RESOURCE_NAME IN ('PROCESSES','SESSIONS');
```

   If the `MAX_UTILIZATION` value is same as the `INITIAL_ALLOCATION` values, then the value of PROCESSES parameter might be too low.

4. Also, check the `listener.log` file for any additional errors. This file is usually located in the `$ORACLE_HOME/network/log` or `$ORACLE_HOME/bin` directory.

   Determine if the ORA-12520 error is followed by this error:

```
TNS-12516: TNS:listener could not find available handler with matching protocol
stack error message
```

   If TNS-12516 is present, the number of connections (threads on Windows systems or processes on UNIX and Linux systems) might have reached the PROCESSES database parameter limit.

5. To resolve this problem, increase the value of the PROCESSES parameter. For example, the following command sets the value to 500:

```
SQL> ALTER SYSTEM SET PROCESSES=500 SCOPE=spfile;
```

   The value of the PROCESSES parameter will depend on your specific environment.

6. Restart the Oracle Database server. For example:

```
SQL> SHUTDOWN IMMEDIATE
SQL> STARUP
```

A.16.1.2.4.2 Diagnosing Listener Control Utility (`lsnrctl`) State

If the Listener Control Utility (`lsnrctl`) shows the state as "blocked" instead of "ready", check for a problem with the listener configuration:

1. Check the value of the LOCAL_LISTENER entry for the Oracle database. For example:

```
# sqlplus ods@oiddb
SQL> SHOW PARAMETER LOCAL_LISTENER
NAME                               TYPE        VALUE
---------------------------------- ----------- ------------------------------
local_listener                     string      LISTENER_ORCL
```

2. Compare the LOCAL_LISTENER value from the previous step with the value in the `tnsnames.ora` file. For example, consider the following the output from the `tnsnames.ora` file:

```
...
LISTENER_ORCL =
(ADDRESS = (PROTOCOL = TCP)(HOST = host.example.com)(PORT = 1521))
```

3. If the listener entry in the `tnsnames.ora` file does not match the LOCAL_LISTENER parameter from the database, configure the LOCAL_LISTENER parameter to point to the listener.

   Update the LOCAL_LISTENER parameter for the database. For example:

```
SQL> ALTER SYSTEM SET LOCAL_LISTENER ='(ADDRESS=(PROTOCOL=TCP)(HOST=host.example.com)
(PORT=1521))' scope=spfile;
```

Or, if you have an entry in your `tnsnames.ora` that points to the listener, use that listener name instead. For example:

```
ALTER SYSTEM SET LOCAL_LISTENER ='listener_name' scope=spfile;
```

A.16.1.3 Directory Server Error Messages and Causes

This section contains a list of Oracle directory server error messages that you might encounter.

Each message is followed by its most probable causes. Also, see OID Error Messages section in *Oracle Fusion Middleware Error Messages Reference*.

- Inappropriate Authentication Error
- Constraint Violation Error Due to Editing a User or Group or Creating a Realm
- Standard Error Messages Returned from Oracle Directory Server
- Additional Directory Server Error Messages

A.16.1.3.1 Inappropriate Authentication Error

You see the following error message on the command line when attempting an anonymous bind to the server:

```
ldap_bind: Inappropriate authentication
ldap_bind: additional info: Server is Configured to Deny Anonymous Binds
```

Problem

Anonymous binds are disabled. In most environments, some clients require anonymous access.

Solution

Enable anonymous binds.

> **See Also:**
>
> Managing Anonymous Binds for more information.

A.16.1.3.2 Constraint Violation Error Due to Editing a User or Group or Creating a Realm

You get the following error in `oidldap*.log`:

```
ORA-01483: invalid length for DATE or NUMBER bind variable.
```

You may also see the following error on your screen:

```
LDAP: error code 19 - Constraint Violation
```

These errors might only occur intermittently.

Problem

If you loaded the OracleAS Metadata Repository into an Oracle 10*g* Database that uses the AL32UTF8 character set, you may encounter some errors when you try to edit a user or Group, or Create Identity Management Realms in Oracle Internet Directory. Editing a user includes editing attributes for an existing user.

Solution

As a workaround, you can wait a bit and try editing the user again.

A.16.1.3.3 Standard Error Messages Returned from Oracle Directory Server

Table A-28 lists standard error messages and their causes. Oracle Internet Directory also returns other messages listed and described in Additional Directory Server Error Messages.

Table A-28 Standard Error Messages

Error	Cause
00: LDAP_SUCCESS	The operation was successful.
01: LDAP_OPERATIONS_ERROR	General errors encountered by the server when processing the request.
02: LDAP_PROTOCOL_ERROR	The client request did not meet the LDAP protocol requirements, such as format or syntax. This can occur in the following situations: Server encounters a decoding error while parsing the incoming request. The request is an add or modify request that specifies the addition of an attribute type to an entry but no values specified. Error reading SSL credentials. An unknown type of modify operation is specified (other than LDAP_MOD_ADD, LDAP_MOD_DELETE, and LDAP_MOD_REPLACE) Unknown search scope
03: LDAP_TIMELIMIT_EXCEEDED	Search took longer than the time limit specified. If you have not specified a time limit for the search, Oracle Internet Directory uses a default time limit of one hour.
04: LDAP_SIZELIMIT_EXCEEDED	More entries match the search query than the size limit specified. If you have not specified a size limit for the search, Oracle Internet Directory uses a default size limit of 1000.
05: LDAP_COMPARE_FALSE	Presented value is not the same as the one in the entry.
06: LDAP_COMPARE_TRUE	Presented value is same as the one in the entry.
07: LDAP_STRONG_AUTH_NOT_SUPPORTED	The requested bind method is not supported by the server. For example, SASL clients requesting Kerberos authentication from Oracle Internet Directory receive this error in response.
09: LDAP_PARTIAL_RESULTS	Server returned a referral.
10: LDAP_REFERRAL	Server returned a referral.
12: LDAP_UNAVAILABLE_CRITICALEXTENSION	Specified request is not supported
16: LDAP_NO_SUCH_ATTRIBUTE	Attribute does not exist in the entry specified in the request.
17: LDAP_UNDEFINED_TYPE	Specified attribute type is undefined in the schema.
19: LDAP_CONSTRAINT_VIOLATION	The value in the request violated certain constraints.
20: LDAP_TYPE_OR_VALUE_EXISTS	Duplicate values specified for the attribute.
21: LDAP_INVALID_SYNTAX	Specified *attribute* syntax is invalid. In a search, the *filter* syntax is invalid.

Table A-28 (Cont.) Standard Error Messages

Error	Cause
32: LDAP_NO_SUCH_OBJECT	The base specified for the operation does not exist.
34: LDAP_INVALID_DN_SYNTAX	Error in the DN syntax.
49: LDAP_INVALID_CREDENTIALS	Bind failed because the credentials are not correct.
50: LDAP_INSUFFICIENT_ACCESS	The client does not have access to perform this operation.
53: LDAP_UNWILLING_TO_PERFORM	General error, or server is in read-only mode.
65: LDAP_OBJECT_CLASS_VIOLATION	A change to the entry violates the object class definition.
66: LDAP_NOT_ALLOWED_ON_NONLEAF	The entry to be deleted has children.
67: LDAP_NOT_ALLOWED_ON_RDN	Cannot perform the operation on RDN attributes—for example, you cannot delete the RDN attribute of the entry.
68: LDAP_ALREADY_EXISTS	Duplicate ADD condition.
81: LDAP_SERVER_DOWN	Cannot contact the directory server. This message is returned from the SDK.
82: LDAP_LOCAL_ERROR	The client encountered an internal error. This message is returned from the client SDK.
83: LDAP_ENCODING_ERROR	The client encountered an error in encoding the request. This message is returned from the SDK.
84: LDAP_DECODING_ERROR	The client encountered an error in decoding the request. This message is returned from the SDK.
85: LDAP_TIMEOUT	Client encountered the time out specified for the operation. This message is returned from the SDK.
86: LDAP_AUTH_UNKNOWN	Authentication method is unknown to the client SDK.
87: LDAP_FILTER_ERROR	Bad search filter
88: LDAP_USER_CANCELLED	User cancelled operation
89: LDAP_PARAM_ERROR	Bad parameter to an LDAP routine
90: LDAP_NO_MEMORY	Out of memory

A.16.1.3.4 Additional Directory Server Error Messages

Table A-29 lists additional directory server error messages and their causes. These messages do not display error codes.

The Oracle Internet Directory application replaces the *parameter* tag seen in some of the following messages with the appropriate run-time value.

Table A-29 Additional Error Messages

Error	Cause
%s attribute not found	The particular attribute type is not defined in the schema.
parameter not found for attribute *parameter*	Value not found in the attribute. (ldapmodify)
Admin domain does not contain schema information for objectclass *parameter*	The object class specified in the request is not present in the schema.
Attempted to add a Class with oid *parameter* taken by other class	Duplicate object identifier specified. (schema modification)
Attribute *parameter* already in use	Duplicate attribute name. (schema modification)
Attribute *parameter* has syntax error.	Syntax error in the attribute name definition. (schema modification)
Attribute *parameter* is not supported in the schema.	Attribute not defined. (all operations)
Attribute *parameter* is single valued.	Attribute is single-valued. (ldapadd and ldapmodify)
Attribute *parameter* not present in the entry.	This attribute does not exist in the entry. (ldapmodify)
Bad attribute definition.	Syntax error in attribute definition. (schema modification)
Currently Not Supported	The version of LDAP request is not supported by this server.
Entry to be deleted not found.	DN specified in the delete operation not found.
Entry to be modified not found	The entry specified in the request is not found.
Error encountered while adding *parameter* to the entry	Returned when modify add operation is invoked. A possible cause is that the system resource is unavailable.
Error encountered while encrypting an attribute value.	Error in encrypting user password. (all operations)
Error in DN Normalization.	DN specified is invalid. Syntax error encountered in parsing the DN. (all operations)
Error in hashing *parameter* attribute.	Error in creating hash entry for the attribute. (schema modification)
Error in hashing *parameter* objectclass.	Error in creating hash entry for the objectclass. (schema modification)
Error in Schema hash creation.	Error while creating hash table for schema. (schema modification)
Error replacing *parameter*.	Error in replacing this attribute. (ldapmodify)
Error while normalizing value for attribute *parameter*.	Error in normalizing value for the attribute. (all operations)
Failed to find *parameter* in mandatory or optional attribute list.	Attribute specified does not exist in either the mandatory or optional attribute list as required by the object class(es).
Function Not Implemented	The feature/request is currently not supported. (Specifying a non-indexed attribute in a search can generate this error.)
INVALID ACI is *parameter*	The particular ACI you specified in a request is invalid.

Table A-29 (Cont.) Additional Error Messages

Error	Cause
Mandatory attribute *parameter* is not defined in Admin Domain *parameter*.	MUST refers to attribute not defined. (schema modification)
Mandatory Attribute missing.	The mandatory attribute for the particular entry is missing, as required by the particular object class.
Matching rule, *parameter*, not defined.	Matching rule not defined in the server. (schema modification)
MaxConn Reached	The maximum number of concurrent connections to the LDAP server has been reached.
Modifying the Naming attribute for the entry without modifying the DN.	Cannot modify the naming attributes using ldapmodify. A naming attribute, such as `cn` is an element in the DN.
New Parent not found.	New parent specified in modifydn operation does not exist. (ldapmodifydn)
Object already exists.	Duplicate entry. (ldapadd and ldapmodifydn)
Object ID *parameter* already in use.	Duplicate object identifier specified. (schema modification)
Objectclass *parameter* already in use.	Duplicate Objectclass name. (schema modification)
Objectclass attribute missing.	The objectclass attribute is missing for this particular entry.
OID *parameter* has syntax error.	syntax error in the object identifier definition. (schema modification)
One of the attributes in the entry has duplicate value.	You entered two values for the same attribute in the entry you are creating.
Operation not allowed on the *parameter*.	Operation not allowed on this entry. (modify, add, and delete)
Operation not allowed on the DSE Entry.	Can't do this operation on DSE entry. (delete)
Optional attribute *parameter* is not defined in Admin Domain *parameter*.	MAY refers to attribute not defined. (schema modification)
Parent entry not found in the directory.	Parent entry does not exist. (ldapadd and perhaps ldapmodifydn)
Super object *parameter* is not defined in Admin Domain *parameter*.	SUP types refer to non-existing class. (schema modification)
Super type undefined.	SUP type does not exist. (schema modification)
Superuser addition not permitted.	Cannot create superuser entry. (ldapadd)
Syntax, *parameter*, not defined.	Syntax not defined in the server. (schema modification)
The attribute or the value specified in the RDN does not exist in the entry.	AVA specified as the RDN does not exist in the entry. (ldapadd)
Unknown search scope	The search scope specified in the LDAP request is not recognized.
Version Not Supported	The version of the LDAP request is not supported by this server.

Table A-29 (Cont.) Additional Error Messages

Error	Cause
Alias Problem	Either of the following have occurred: • An alias was dereferenced, but it did not point to an entry in the DIT. • The user tries to add an alias entry whose parent is an alias.
Alias Dereferencing Problem	The user cannot dereference an alias because of access control issues.
No Such Object	The server cannot find the base DN specified in the search request.
Invalid DN Syntax	When adding or modifying an alias entry, if the value specified for `aliasedObjectName` has invalid DN syntax, then the directory server returns this error message to the client.
Insufficient Access Rights	The user does not have access to the dereferenced entry.

A.16.1.4 Core Dump and Stack Trace Occurs When Oracle Internet Directory Crashes

You can control the type of information Oracle Internet Directory provides when it crashes by changing the value of the `orclsdumpflag` attribute in the instance-specific configuration entry.

If the server crashes, it leaves a core file under the directory

$DOMAIN_HOME`/tools/OID/logs`

If `orclsdumpflag` is set to `0`, and the server crashes, in addition to the core dump, the server also attempts to leave a stack trace. The location for the stack trace is:

$DOMAIN_HOME`/servers/OID/logs/`*componentName*`/oidldapd_stack00_`*pid*`.dmp`

Some operating system-specific settings can affect the generation of a core dump or stack trace. Consult your operating system documentation to determine whether the following settings are required:

- The `coredump` parameter must be set to allow core dumps.
- The file size limit, as specified with the `ulimit` command, must be sufficient to allow core dumps.
- The file permissions on the *$ORACLE_HOME*`/bin/oidldapd` binary file must allow read by group. You can ensure that group has read permission by typing:

  ```
  chmod g+r $ORACLE_HOME/bin/oidldapd
  ```

  as the root user.

A.16.1.5 TCP/IP Problems

TCP/IP bugs in the operating system can interfere with Oracle Internet Directory service.

Refer to the following topics:

- Do Not Use TCP-Based Monitoring of Server Availability on Windows 2003 Server
- Do Not Install DaimondCS Port Explorer

A.16.1.5.1 Do Not Use TCP-Based Monitoring of Server Availability on Windows 2003 Server

If you use the F5 load balancer for monitoring Oracle Internet Directory server availability, configure the load balancer to use LDAP- or HTTP-based monitoring, as described in Configuring Oracle HTTP Server for High Availability in the *Oracle Fusion Middleware High Availability Guide* . Using TCP-based monitoring might cause the service to become unavailable, due to an operating system bug on Microsoft Windows 2003 Server.

A.16.1.5.2 Do Not Install DaimondCS Port Explorer

Oracle Internet Directory does not work if DaimondCS Port Explorer is installed on the system.

A.16.1.6 Troubleshooting Password Policies

This section describes error messages and problems related to password policies.

This section includes the following:

- Password Policy is Not Enforced
- Password Policy Error Messages

A.16.1.6.1 Password Policy is Not Enforced

The password policy is not being enforced for a given user or set of users. For example, users can reset their password using a syntax that is disallowed by the defined password policy.

Problem

Just creating a password policy is not sufficient. You must also specify the subtree to be governed by the policy.

Solution

Add and populate a `pwdPolicysubentry` attribute with the policy's DN, at the root of that subtree.

> **See Also:**
>
> Creating and Applying a Password Policy for more information.

A.16.1.6.2 Password Policy Error Messages

Table A-30 contains the error messages sent to the client as a result of password policy violations. The error codes are not standard LDAP error codes. They are messages sent as a part of additional information in the LDAP result.

Table A-30 Password Policy Violation Error Messages

Error Number	Exception	Comment or Resolution
9000	`GSL_PWDEXPIRED_EXCP`	User's password has expired.
9001	`GSL_ACCOUNTLOCKED_EXCP`	User account is locked.

Table A-30 (Cont.) Password Policy Violation Error Messages

Error Number	Exception	Comment or Resolution
9002	GSL_EXPIREWARNING_EXCP	User password will expire in pwdexpirewarning seconds. Please change your password now.
9003	GSL_PWDMINLENGTH_EXCP	User password is not the required number of characters long.
9004	GSL_PWDNUMERIC_EXCP	User password does not contain required numeric characters.
9005	GSL_PWDNULL_EXCP	User password is a null password, which is disallowed.
9006	GSL_PWDINHISTORY_EXCP	User's new password is the same as an old one saved in history, which is disallowed.(The pwdinhistory attribute controls the number of passwords saved in history.)
9007	GSL_PWDILLEGALVALUE_EXCP	The user password supplied is an illegal value defined in orclpwdillegalvalues, and therefore cannot be used.
9008	GSL_GRACELOGIN_EXCP	User password has expired. User has pwdgraceloginlimit grace logins left or orclpwdgracelogintimelimit seconds in which grace logins are allowed.
9012	GSL_PWDALPHA_EXCP	Your Password must contain at least orclpwdminalphachars alphabetic characters.
9013	GSL_PWDSPECIAL_EXCP	Your Password must contain at least orclpwdminspecialchars special characters.
9014	GSL_PWDUPPER_EXCP	Your Password must contain at least orclpwdminuppercase upper case characters
9015	GSL_PWDMAXCHAR_EXCP	Your Password can only contain orclpwdmaxrptchars repeated characters
9016	GSL_PWDLOWER_EXCP	Your Password must contain at least orclpwdminlowercase lowercase characters.
9017	GSL_EC_PWDPOLSUBENTINV	The pwdPolicysubentry provided is invalid. (The DN is not that of a valid password policy present in the directory.)
9018	GSL_EC_PWDPOLINUSE	The pwdPolicy entry you are deleting is currently in use. (You must remove the references to the password policy before the policy itself can be removed.)
9019	GSL_EC_PWDPOLOBJ	The DN of a pwdPolicy entry may not be modified.
9020	GSL_PWDMINAGE_EXCP	Your Password has to be at least pwdminage seconds old before it can be changed.
9032	GSL_EC_GRACE_CONST	orclgracelogintimelimit and pwdgraceloginlimit are mutually exclusive. Both cannot be nonzero.
9033	GSL_EC_NOROOTDSEPWDPOL	The pwdpolicysubentry attribute in the root DSEcannot be deleted. (This is not allowed because it would leave the directory without an applicable password policy.)
9034	GSL_EC_NOTROCPWDPOL	Only password policies defined in the Root Oracle Context are applicable in the Root DSE. (This ensures that only a policy specified by an admin who has directory-wide privileges can be applied to the entire directory.)
9050	GSL_ACCTDISABLED_EXCP	User account has been disabled.

> **See Also:**
>
> Managing Accounts and Passwords in Oracle Internet Directory.

A.16.1.7 Troubleshooting Directory Performance

This section gives some quick pointers for common performance-related problems.

Refer to the following topics:

- Poor LDAP Search Performance
- Improving Poor LDAP Add or Modify Performance
- Diagnosing Poor Oracle Database Server Performance
- Troubleshooting Database Performance Issues Using AWR Reports

A.16.1.7.1 Poor LDAP Search Performance

LDAP search performance is poor.

Problem

Various problems.

Solution

Ensure that:

- Schema associated with the `ODS` user is `ANALYZED`
- For searches involving multiple filter operands, make sure that the order in which they are given goes from the most specific to the least specific. For example, `&(uid=john.doe)(objectclass=person)` is better than `&(objectclass=person)(uid=john.doe)`.

Also see Diagnosing Poor Oracle Database Server Performance.

A.16.1.7.2 Improving Poor LDAP Add or Modify Performance

LDAP add or modify performance is poor.

Problem

Various problems

Solution

Ensure that:

- There are enough redo log files in the database
- The undo tablespace in the database is large enough
- The schema associated with the `ODS` user is `ANALYZED`

When estimating the statistics, you can use the OID Database Statistics Collection tool to analyze the various database ODS schema objects.

Both the tracing functionality described in Managing Logging and the database tracing event 10046 can assist you in diagnosing performance issues.

> **See Also:**
>
> The `oidstats.sql`command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management*for instructions on using the OID Database Statistics Collection tool
>
> The Oracle Internet Directory Performance Tuning in *Oracle Fusion Middleware Performance and Tuning Guide* for instructions on optimizing searches
>
> Note 243006.1 on My Oracle Support (formerly MetaLink), https://support.oracle.com, for information on performance issues with group entries

A.16.1.7.3 Diagnosing Poor Oracle Database Server Performance

Problem

Oracle database server is consuming lot of processor resources during LDAP search operations.

Solution

Proceed as follows:

1. Identify the LDAP operations that are processor-intensive by running:

   ```
   oidctl connect=connstr status -diag
   ```

   This command displays the LDAP operation and associated SQL that is being executed.
2. Tune the database appropriately for this kind of query. See the Basic Tuning Considerations chapter in *Oracle Fusion Middleware Performance and Tuning Guide*.
3. If possible, change the applications's search signature. If that is not possible, tune the Oracle Internet Directory attribute `orclinmemfiltprocess`. See the Oracle Internet Directory chapter in Basic Tuning Considerations chapter in *Oracle Fusion Middleware Performance and Tuning Guide*.

A.16.1.7.4 Troubleshooting Database Performance Issues Using AWR Reports

The Automatic Workload Repository (AWR) is a built-in repository that exists in Oracle Database. At regular intervals, the database makes a snapshot of its vital statistics and workload information and stores this snapshot in the workload repository.

By default, Oracle database automatically generates snapshots of the performance data once every hour and retains the statistics in the repository for eight days.

If you suspect Oracle Internet Directory as the component causing a database performance issue, you can use AWR reports to further investigate the problem, as described in the following sections:

- Generating an AWR Report
- Analyzing an AWR Report

A.16.1.7.4.1 Generating an AWR Report

To generate a specific AWR report, run the `awrrpt.sql` SQL script:

1. Connect to the Oracle database as the `sysdba` user. For example:

```
sqlplus /sys/sysdba-password@oiddb as sysdba
```

2. Generate the AWR report by running the `awrrpt.slq` script at the SQL prompt. For example:

```
SQL> $ORACLE_HOME/rdbms/admin/awrrpt.sql
```

3. When prompted, specify the following information for the report:
   - HTML or text format. The default format is HTML.
   - Number of days for which you want to list snapshot IDs.

     This value should indicate the time period in days when the performance issue occurred. For example, if the issue occurred today, then enter 1. If the issue occurred yesterday, enter 2.

     The `awrrpt.sql` script lists snapshot IDs for the number of days selected. Each snapshot is assigned a unique snapshot ID.
   - Beginning and ending snapshot ID for the workload repository report.

     Identify the time when the performance issue occurred. Based on this time period, define the range of snapshots for which AWR data needs to be extracted by specifying a beginning and ending snapshot ID. For example:

     ```
     Enter value for begin_snap: 110
     Enter value for end_snap: 120
     ```
   - Report name. Accept the default report name, or enter a new name.

   The database generates the AWR report in the directory where you executed the `sqlplus` command.
4. Analyze the AWR report to try to determine the issue causing the performance issue, as described in the next section.

A.16.1.7.4.2 Analyzing an AWR Report

To analyze an AWR report:

1. Check the "Top 5 Timed Foreground Events" table. This table can provide information about events that might be causing performance issues. Check the "Waits" and the "Avg wait (ms)" columns and note the event with the highest average wait time.
2. In the "SQL Statistics" section, check the "SQL ordered by Elapsed Time" table:
   a. Check the values in the "Elapsed Time per Exec (s)" column.

      Under normal conditions, a query should execute in less than one second (shown in milliseconds in the table). If a query is taking a second or more to execute, it should be investigated in the next step.
   b. For the event that displays the highest "Elapsed Time per Exec (s)", click the value under the "SQL Id" column to show the SQL query that was executed.

      If this SQL query has fields such as `"attrvalue = 'referral'"`, the query can be skipped from being executed in environments where only one Oracle Internet Directory instance is running and there are no referrals to other Oracle Internet Directory processes.
3. If there are no referral entries in your environment, you can improve performance by preventing the previous query from being executed, as follows:
   a. In `"cn=dsaconfig,cn=configsets,cn=oracle internet directory"`, set the `orclskiprefinsql` attribute to 1, to skip the referral for the search.

For more information, see Attributes of the DSA Configuration Entry.

b. Restart Oracle Internet Directory for the change to take effect.

A.16.1.8 Troubleshooting Port Configuration

You can find out which ports the Oracle Internet Directory dispatcher is using for SSL and non-SSL connections in the following ways:

- In Oracle Enterprise Manager Fusion Middleware Control, select **Port Usage** from the OID menu.
- From the command line, execute:

```
oid_instanceStatus(instanceName = 'instance-name')
```

- From the command line, execute:

```
oidctl connect=oiddb status
```

A.16.1.9 Troubleshooting Starting Oracle Internet Directory

This section describes problems you might encounter when starting Oracle Internet Directory.

Refer to the following topics:

- Oracle Internet Directory is Down
- Oracle Internet Directory is Read-Only

A.16.1.9.1 Oracle Internet Directory is Down

Problem

Oracle Enterprise Manager Fusion Middleware Control shows Oracle Internet Directory down. The command:

```
oid_instanceStatus(instanceName = 'instance-name')
```

shows that `oidmon` is down, as well as all the `oidldapd` processes.

Solution

Consult the `oidmon` log, *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName/oidmon-0000.log`* and `nodemanager` log, *`$DOMAIN_HOME`* `/nodemanager/nodemanager.log` to determine why oidmon is not starting.

Problem

Oracle Enterprise Manager Fusion Middleware Control shows Oracle Internet Directory down. The command:

```
oid_instanceStatus(instanceName = 'instance-name')
```

shows that `oidmon` is up, but the `oidldapd` processes are down.

Solution

Check the following logs in the order shown:

1. The `oidmon` log, *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName`*`/oidmon-0000.log` contains details as to why `oidmon` cannot start the `oidldapd` process. The most common issues are
   - Unable to connect to Oracle Database: Ensure that the Oracle database and listener are up and running.
   - Time difference between the two nodes is more than 250 seconds: Adjust the system time.
   - `Oidmon` keeps trying to start `oidldapd` processes, but they fail to run. To debug, see Step 2.
2. The Oracle Internet Directory dispatcher log, *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName`*`/oidldapd01-0000.log` contains information about why `oidldapd` server processes fail to start. The most common reasons are:
   - Configured PORT for Oracle Internet Directory is not free: Execute

     ```
     netstat -an | grep oidPort
     ```

     to see if it the port is free.
   - Oracle Internet Directory is configured to listen on a port number less than 1024 on a UNIX or Linux system and the executable binary file *`$ORACLE_HOME`*`/bin/oidldapd` is either not owned by `root` or does not have the `setuid` bit set.
   - The `oidldapd` dispatcher keeps spawning `oidldapd` server processes, but they fail to run. In this case, you might see a single `oidldapd` dispatcher process running if you use `ps` on UNIX or Linux or Task Manager on Windows.
3. The Oracle Internet Directory server log, *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName`*`/oidldapd01s`*`PID`*`-0000.log` contains information about why the server processes fail to run. Common issues include:
   - Unable to create Oracle Database connection pool: Check the Oracle Database `PROCESSES` parameter and increase if necessary.
   - Oracle Internet Directory is configured to use an SSL wallet file, and that file is inaccessible.

A.16.1.9.2 Oracle Internet Directory is Read-Only

Problem

The Oracle Internet Directory server starts in read-only mode.

Solution

This usually indicates that the Oracle Internet Directory server has been started against the wrong schema. To verify, type these two commands:

```
oidldapd -v

ldapsearch -p oidPort -D cn=orcladmin -q -b "" -s base "objectclass=*"
Orcldirectoryversion
```

If these commands show different versions, the server starts in read-only mode.

A.16.1.10 Oracle Internet Directory Error Due to Interrupted Client Connection

This section provides information on fixing the errors that may occur due to interrupted client connection in Oracle Internet Directory.

You get the following error:

```
ServerDispatcher:  sgslufread: Hard error on read, OS error = 110
```

Problem

This error indicates that an LDAP client has disconnected abruptly. Possible causes include:

- The client machine crashed or the client program terminated the connection without performing an unbind or abandon.
- A network component, such as a load balancer or firewall, broke the connection due to a configured timeout setting such as the idle connection timeout.
- The load balancer issues a TCP ping instead of an LDAP ping.

Solution

This error is due to conditions external to the server. If necessary, inform the network administrator.

A.16.1.11 Troubleshooting Starting, Stopping, and Restarting of the Directory Server

To troubleshoot starting and stopping the directory server, you must know the purpose of each tool involved, how all the tools work together, and the overall process for starting and stopping the server.

This section includes the following topics:

- Starting, Stopping, and Restarting the Directory Server Instance Using the Tool
- Problems Starting, Stopping, and Restarting the Directory Server

> **See Also:**
>
> The Oracle Internet Directory Performance Tuning chapter in *Oracle Fusion Middleware Performance and Tuning Guide*.

A.16.1.11.1 Starting, Stopping, and Restarting the Directory Server Instance Using the Tool

You start the directory server instance by typing:

```
$DOMAIN_HOME/bin/startComponent.sh <instance-name>
```

OIDCTL

When OIDCTL is executed, it connects to the database as user `ODS`. Depending on the options used in the command, it either inserts or updates rows into a table named `ODS.ODS_PROCESS_STATUS`_STATUS. If the `START` option is used, then a row is inserted. If either the `STOP` or `RESTART` option is used, then a row is updated.

The ODS.ODS_PROCESS_STATUS table includes the following information:

- `instance`: The unique number of the instance, any value between 0 and 1000
- `pid`: Process identifier, which is updated by OIDMON when the process is started
- `state`: The type of operation requested

  The possible values for `state` are:

  - 0=stop
  - 1=start
  - 2=running
  - 3=restart
  - 4=shutdown
  - 5=failedover

OIDMON

To start, stop, or restart a directory server instance, OIDMON must be running. At specified intervals, this daemon checks the value of the `state` column in the `ODS.ODS_PROCESS_STATUS` table.

If it finds a row with `state=0`, then it reads the `pid` and stops the process.

If it finds one with `state=1` or `state=4`, then it starts a new process and updates the pid column with a new process identifier.

If it finds one with `state=2`, then it reads the `pid` and verifies that the process with that `pid` is running. If it is not running, then OIDMON starts a new process and updates the `pid` column with a new process identifier.

If it finds a row with `state=3`, then OIDMON reads the `pid`, stops the process, starts a new one, and updates the `pid` accordingly.

If it is unsuccessful, it pushes the request to another node.

In short, OIDCTL inserts and updates state information in the rows in the `ODS.ODS_PROCESS_STATUS` table. OIDMON then reads that information and performs the specified task.

About the Processes Involved in Starting, Stopping, and Restarting the Directory Server

Starting, stopping and restarting the directory server involves processes. OIDMON is one process. On UNIX, it is called `oidmon`. In a Microsoft Windows environment, it is called `oidmon.exe`.

To start an instance, OIDMON checks the unique number in the `instance` column mentioned in the previous section. It then starts another process, namely, the listener/dispatcher, which is different from the Oracle Net Services listener process. It stores the process identifier for that new process in the `pid` column.

The listener/dispatcher, in turn, starts a number of server processes as defined in the configuration set entry. Note that these server processes are controlled by the listener/dispatcher and not by OIDMON. If one of these processes fails, then it is automatically restarted by the listener/dispatcher.

Together, the listener/dispatcher and the server processes constitute a directory server instance. On UNIX, this directory server instance is called `oidldapd`. On Microsoft Windows, they are called `oidldapd.exe`.

In short, there are at least three processes: one for OIDMON and at least two for the directory server itself. When all processes are running, you should see something like the following on UNIX computers:

```
% ps -ef|grep oid
root 12387 12381 0 Mar 28 ? 0:05 oidldapd -i 1 -conf 0 key=811436710
root 12381 1 0 Mar 28 ? 0:10 oidmon start
root 13297 1 0 Mar 28 ? 0:14 oidldapd
```

Another way to obtain server information is by running:

```
oidctl connect=oiddb status.
```

A.16.1.11.2 Problems Starting, Stopping, and Restarting the Directory Server

This section describes some problems you might have when starting, stopping, or restarting the directory server.

A.16.1.11.2.1 OIDCTL or OIDMON fails

Either OIDCTL or OIDMON can fail for reasons.

Problem

Incorrect syntax

Solution

Verify that you are using the correct syntax as described in Oracle Internet Directory Administration Tools *Oracle Fusion Middleware Reference for Oracle Identity Management*. Note that the correct value of the connect option when using OIDCTL is the TNS alias—that is, the connect string—and not a host name or other value.

Problem

The Oracle Internet Directory-designated database is not running.

The Oracle Net Services configurations are incorrect.

Solution

Verify that the Oracle Internet Directory-designated database and the Oracle Net Services components are correctly configured and running. To do this, see if you can connect to the database by using SQL*Plus that is installed in the same *ORACLE_HOME* as OIDCTL. Log in as `ODS`/*ods_password*`@`*tns_alias* where *tns_alias* is the same as that used in the `connect` option with OIDCTL.

Problem

Missing `oidldapd` file.

Solution

See *$DOMAIN_HOME*`/servers/OID/logs/`*componentName*`/oidmon-XXXX.log`. Look for the message: `No such file or directory`. To correct the problem, replace the executable file.

Problem

Wrong permissions on `oidldapd`/oiddispd executable file.

Solution

Part 1

This solution is applicable for Oracle Internet Directory pre 11*g* Release 1 (11.1.1.7.0) release.

Look for the message `Exec of OIDLDAPD failed with error 13`. On UNIX, the `$ORACLE_HOME/bin/oidldapd` file must have the following permissions:

```
-rws--x---  1 root   dba       1691802 Jan 20 10:30 oidldapd
```

If the permissions are not correct, type the following, as root:

```
cd $ORACLE_HOME/bin
chown root:dba oidldapd
chmod 0710 oidldapd
chmod u+s oidldapd
```

Part 2

This solution is applicable for Oracle Internet Directory 11*g* Release 1 (11.1.1.7.0) release and later.

Look for the message `Exec of OIDDISPD failed with error 13`. On UNIX, the `$ORACLE_HOME/bin/oiddispd` file must have the following permissions:

```
-rws--x---  1 root   dba       1691802 Jan 20 10:30 oiddispd
```

If the permissions are not correct, you can perform either of the following options:

A) Run the following, as root:

```
cd $ORACLE_HOME
oidRoot.sh
```

Or

B) Type the following, as root:

```
cd $ORACLE_HOME/bin
chown root:dba oiddispd
chmod 0710 oiddispd
chmod u+s oiddispd
```

It is recommended that you verify the ownership and permission are set correctly with the following command after applying the preceding solution:

```
$ORACLE_HOME/bin/ls -al oiddispd
```

Problem

You are running as a user with insufficient privilege

Solution

To confirm that this is the problem, see *`$DOMAIN_HOME`*`/servers/OID/logs/`*`componentName/oidmon-XXXXX.log`*.

Look for the message: `Permission denied` or `Open Wallet failed`. This happens if you are not running either as `root` or as the user who is in the `dba` group. To correct the problem, try again as the correct user.

Problem

A port is in use.

Solution

See

$DOMAIN_HOME/servers/OID/logs/*componentName*/*oidldapd00sPID*-XXXX.log.

Look for the message: `Bind failed on...` This indicates that the port that `oidldapd` is configured to listen on is in use by some other process. To determine which process is using the port, type:

```
netstat -a | grep portNum
```

If necessary, reconfigure the other process to use a different port or configure `oidladapd` to listen on another port by adding a configset. Remember that, by default, `oidladapd` listens on two ports, an SSL and non-SSL port.

Problem

On a cluster or Oracle Application Server Cluster (Identity Management) configuration, OIDMON pushes the server to another node in a cluster when it cannot start the server on the local node.

Solution

See `oidmon.log`. Look for the message: `gslsgfrPushServer: Could not start server`*on NodeA*, `trying to start on node`*NodeB*. To correct this problem, you must first determine why OIDMON cannot start the server on the local node.

Problem

A possible problem with Oracle Net Services or with the database itself.

Solution

See `oidmon.log`, `oidldapd`*xx*`.log`, where *xx* is the server instance number.

A Row is Missing from ODS.ODS_PROCESS_STATUS

Problem

In a cluster or Oracle Application Server Cluster (Identity Management) configuration, OIDMON successfully starts `oidldapd` on both nodes, but then initiates failover due to a time stamp difference.

Solution

See the trace files `oidldapd`*xx*`.log` where *xx* is the instance number, and `oidldapd`*xx*`s`*yy*`.log` where `xx` is the instance number and *yy* is the process identifier. If the trace files do not give useful information or pointers to My Oracle Support (formerly MetaLink) documents, then do the following: (1) Stop the directory server processes; (2) Remove or rename old trace files; (3) Start OIDMON and a directory server with maximum debug level, namely, 11744051. Note that, to get the trace files, you must first stop, then start, the server; you cannot simply restart it. Investigate the new trace files, and, if needed, log an iTAR with Oracle Support Services and upload the trace files to the iTAR.

> **See Also:**
>
> The `oidctl` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for more information on failover.

A.16.1.12 Troubleshooting Oracle Internet Directory Replication

This section discusses directory replication problems.

This section includes the following topics:

- Bootstrapping Fails
- Log Files to Diagnose Replication Issues
- Replication Server Does Not Start
- Additional Directory Server Error Messages

A.16.1.12.1 Bootstrapping Fails

Disable referential integrity during the replication bootstrapping process. If referential integrity is enabled, bootstrapping fails.

A.16.1.12.2 Log Files to Diagnose Replication Issues

Whenever you investigate a replication problem, be sure to consult the log files for information. The log files are *$DOMAIN_HOME*/servers/OID/logs/componentName/`oidrepld-XXXX.log`, `oidldapd00-XXXX.log` and *$DOMAIN_HOME*/servers/OID/logs/componentName/`oidldapd00s`*PID*`-XXXX.log` where `PID` is the server process identifier and *XXXX* is a number from 0000 to `orclmaxlogfiles configured`.

The replication server supports multiple debugging levels. To turn on replication debugging, use either `ldapmodify` or the Shared Properties, Replication tab, in Oracle Enterprise Manager Fusion Middleware Control to change `orcldebuglevel` in the replication configuration set.

> **Note:**
>
> Turning on debugging affects replication performance.

> **See Also:**
>
> Managing Replication Configuration Attributes for more information.

A.16.1.12.3 Replication Server Does Not Start

There are several problems that can prevent the replication server from starting.

Problem

Invalid `oidctl` syntax

Solution

Use the following syntax to start the replication server.:

```
oidctl server=oidrepld connect=connect string instance=instance_number \
       flags="-h host -p port"
```

Problem

Oracle Internet Directory is not running at the host and port you specified on the command line when you attempted to start the replication server. This caused the anonymous bind to the target Oracle Internet Directory to fail.

Solution

Make sure the target Oracle Internet Directory is up and running at the specified host and port.

Problem

The replication server is attempting to bind to the host and port specified in either the `orclreplicaprimaryurl` or the `orclreplicasecondaryurl` attribute of the Replica entry, but Oracle Internet Directory is running at a different host or port.

Solution

If you decide to run Oracle Internet Directory at a different host or port, add the new information to the `orclreplicasecondaryurl` attribute of the replica entry, as follows:

1. Prepare a modification file, `mod.ldif`. For example, to change to host my.us.example.com and port 4444, you would specify:

   ```
   dn: orclreplicaid=replica_ID, cn=replication configuration
   changetype: modify
   add: orclreplicasecondaryurl
   orclreplicasecondaryurl: ldap://my.us.example.com:4444/
   ```

2. Run:

   ```
   ldapmodify -h host -p port -f mod.ldif
   ```

Problem

The `ReplBind` credential in the replication wallet *`$DOMAIN_HOME`*`/config/fmwconfig/components/OID/admin/oidpwdr`*`ORACLE_SID`* is corrupt or invalid. That is, the password stored in the wallet is not the same as the password that is stored in the directory, or the wallet does not exist. This causes the replication bind to fail and the replication server to exit with an error.

You might see messages similar to this example in the file `oidrepld`*`XX`*`.log`:

```
2005/07/21:11:13:28 * gslrcfdReadReplDnPswd:Error reading repl passwd
2005/07/21:11:13:28 * gslrcfcReadReplConfig:Error found.
2005/07/21:11:13:28 * Failed to read replication configuration information.
```

Solution

Use `remtool` to fix the replication bind credential in the replication wallet or to synchronize between Oracle Internet Directory and the replication wallet.

- `remtool -pchgpwd` changes the password of the replication dn of a replica. Use this option if you know the current replication DN password stored in the directory and you want to change it both in the directory and in the wallet.

- `remtool -presetpwd` resets the password or the replication dn of a replica. Use this option if you know the current replication DN password stored in the directory and you want to change it both in the directory and in the wallet.
- `remtool -pchgwalpwd` changes password of replication dn of a replica only in the wallet. Use this option if you know the replication DN password stored in the directory but you are not sure whether the wallet has the correct password or you want to create the wallet file.

All of these options create a wallet if one does not already exist.

> **See Also:**
>
> - The`remtool` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for more information about using `remtool`
> - The `oidpasswd` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for more information about using `oidpasswd`

Problem

The replication server is attempting to bind to an SSL port that is configured for one-way or two-way authentication.

Solution

Configure the replication server to use either the non-SSL port or an SSL port configured for no authentication. You can use a separate Oracle Internet Directory server instance just for replication.

> **See Also:**
>
> Use of SSL Encryption in Oracle Internet Directory Replication

A.16.1.12.4 Errors in Replication Bootstrap

errors can occur in replication bootstrap.

Problem

Some of the naming contexts failed to be bootstrapped.

Solution

Identify the naming contexts that failed to be bootstrapped, and use the `oidcmprec` tool to reconcile them.Then resume replication by setting the consumer's replica state to ONLINE mode

Problem

Various causes.

Solution

Identify the cause of the bootstrap failure and fix the cause, then restart bootstrapping by setting consumer's replica state to BOOTSTRAP mode.

Solution

To determine the exact cause of the error, examine the log file `oidldapdxx.log`. Look for error messages like those in the following example:

```
2004/09/14:12:57:23 * Starting OIDREPLD against dlsun1418:4444...
2004/09/14:12:57:25 * Starting scheduler...
2004/09/14:12:57:26 * Start to BootStrap from supplier=dlsun1418_replica to
consumer=dlsun1418_replica2
2004/09/14:12:57:27 * gslrbssSyncDIT:Replicating namingcontext=cn=oraclecontext ......
2004/09/14:12:58:21 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oraclecontext, 222 entries matched
2004/09/14:12:58:21 * gslrbssSyncDIT:Replicating namingcontext=cn=joe smith ......
2004/09/14:12:58:23 * BootStrap failure when adding DN=cn=Joe Smith,
server=dlsun1418_replica2,err=Constraint violation.
2004/09/14:12:58:23 * gslrbssSyncDIT:Sync failed for namingctx: cn=joe smith, only 1
entries retrieved
2004/09/14:12:58:23 * gslrbssSyncDIT:Replicating
namingcontext=cn=oracleschemaversion ......
2004/09/14:12:58:25 * gslrbssSyncDIT:Sync done successfully for namingctx:
cn=oracleschemaversion, 10 entries matched
2004/09/14:12:58:51 * gslrbsbBootStrap: Failure occurred when bootstrapping 1 out of 3
namingcontext(s) from the supplier
```

Identify the cause of the bootstrap failure and fix it. You can identify the naming contexts that caused the problem, then use `oidcmprec` to compare and reconcile the naming contexts. After you resolve the problem, start bootstrapping again by starting the Oracle Internet Directory replication server.

Problem

The Oracle Internet Directory server was shut down during the bootstrapping

Solution

Make sure both the supplier Oracle Internet Directory and the consumer Oracle Internet Directory servers are up and running during replication bootstrapping.

Problem

Some of the entries being bootstrapped cannot be applied at the consumer due to a constraint violation.

Solution

Make sure the Oracle Internet Directory schema of the consumer are synchronized with those of the supplier before starting replication bootstrap. When you add an LDAP replica, `remtool` ensures that the Oracle Internet Directory schema on the consumer replica are synchronized with those on the supplier replica.

Problem

Improper replication filtering during bootstrapping. Replication supports excluding one or more attributes during bootstrapping. However, if a mandatory attribute of an entry is configured to be excluded, that entry cannot be applied at the consumer due to an objectclass violation.

Solution

Follow the replication naming context configuration rules in Setting Up Replication to configure replication filtering properly.

If you are debugging LDAP replication, you should become familiar with the LDAP replica states. If LDAP-based replication is configured, when the replication server starts, it reads the replica state from the local replica. The replication server behaves differently, depending upon the local replica state. LDAP replication errors appear in `oidldapdxx.log`

> **See Also:**
>
> LDAP Replica States

Problem

When you restart the replication server after the replication server failed to bootstrap a naming context having more than 5000 entries, you may see error messages similar to this in the log file `oidrepld00.log`:

```
2005/04/05:13:21:55 * gslrbssSyncDIT:Replicating namingcontext=dc=com ......
2005/04/05:15:36:09 * gslrbssSyncDIT:Subtree delete on dc=com failed.
Error=DSA is unwilling to perform
2005/04/05:15:36:09 * gslrbssSyncDIT:Sync failed for namingctx: dc=com, only
0 entries retrieved
```

The replication server performs two steps during bootstrap operation. First, in the consumer, it deletes the naming contexts that it has to bootstrap. Second, it copies entries belonging to those naming contexts from supplier to consumer. Deletion by the replication server of a naming context having several thousands of entries results in a big transaction. The undo tablespace must have sufficient space to accommodate a big transaction. If the database's undo tablespace does not have sufficient space, it results in an ORA-30036 error.

Solution

Either have the database administrator add more space to the undo tablespace, or use the `bulkdelete` tool to delete the required naming context before you start the replication server.

A.16.1.12.5 Changes Are Not Replicated

Changes are not replicated from one node to another.

Problem

The replication server has run out of table space

Solution

Look for the following message in the server log:

```
OCI Error ORA-1653 : ORA-01653: unable to extend table ODS.ASR_CHG_LOG by 8192 in
tablespace OLTS_DEFAULT
```

Extend the table space and investigate why the table space keeps growing.

Problem

The target Oracle Internet Directory server is down.

Solution

Restart the target Oracle Internet Directory server.

Problem

Various causes

Solution

Make sure the replication server is started on all nodes, in multi-master replication, and at the consumer node in single-master or fan-out replication.

Check the replication log and LDAP log for error messages and fix the cause of the error after investigation.

> **See Also:**
>
> The `remtool` command-line tool reference in *Oracle Fusion Middleware Reference for Oracle Identity Management* for more information about using `remtool`.

A.16.1.12.6 Replication Stops Working

Problem

Data is not replicated between the replicas. In some cases, a working replication setup stops working after OID Human Intervention Queue entries are applied to one of the nodes. In other cases, adding or deleting a new replica causes problems or failures.

Problem

Various causes

Solution

See the following Notes on My Oracle Support (formerly MetaLink), https://support.oracle.com:

Note 171693.1, "Resolving Conflicts"

Note 213910.1, "Debugging OID Replication when ASR_CHG_LOG Never Gets Populated."

You can search for Notes by entering a term such as "replication" into the search box.

A.16.1.13 Troubleshooting Change Log Garbage Collection

Both replication and Oracle Directory Integration Platform use change logs to propagate information from a supplier directory to a consumer directory. All change logs are stored in the table `ods_chg_log`. In addition, replication change logs are stored in `asr_chg_log`.

This section discusses possible problems you might encounter with change log garbage collection.

Problem

Garbage collection is not working and Oracle Internet Directory is using Oracle Database 11.2.0.1.

Solution

Apply 11.2.0.1.3 PSU to the database.

Problem

Change logs are not being purged due to a replication issue. For example, if a replication server has been down for a few days, replication change logs are not purged because they are needed for replication recovery.

Solution

Resolve the replication issue. See "Troubleshooting Oracle Internet Directory Replication"".

Problem

The attribute `orclpurgetargetage` is set too high and there are one or more enabled but inactive change log subscribers that do not update `orclLastAppliedChangeNumber` in their subscriber profiles. Change number-based purging won't purge change logs that are not yet consumed and time-based purging won't purge them because they're not old enough.

Solution

Set the attribute `orclpurgetargetage`to a smaller value so that change logs are purged sooner.

Solution

Disable inactive changelog subscribers so that change logs are purged by change log number-based purging. Locate such enabled but inactive subscriber profiles by examining the `orclLastAppliedChangeNumber` in all subscriber profiles by typing:

```
ldapsearch -v -p port -h host -D cn=orcladmin -q \
           -b "cn=changelog subscriber,cn=oracle internet directory" \
           -s sub "objectclass=orclchangesubscriber" \
           orcllastappliedchangenumber orclsubscriberdisable
```

Look for an entry that has `orclSubscriberDisabled` equal to zero and an `orclLastAppliedChangeNumber` value that never changes. If such an entry exists, and the change log garbage collector's `orclpurgetargetage` is zero or greater, delete the value of `orclpurgetargetage`. When `orclpurgetargetage` is not defined or less than zero, the garbage collector purges changes applied by the replication server, even if another subscriber has not updated its `orclLastAppliedChangeNumber`.

> **See Also:**
>
> "Overview of Change Log Purging".

A.16.1.14 Troubleshooting Dynamic Password Verifiers

This section lists and describes the ways to troubleshoot the error messages for dynamic password verifiers.

Table A-31 lists and describes the error messages for dynamic password verifiers.

Table A-31 Error Messages for Dynamic Password Verifiers

Error Code	Description
9022	A reversible encrypted password is missing from the user entry.
9023	The crypto type specified in the LDAP request control is not supported.

Table A-31 (Cont.) Error Messages for Dynamic Password Verifiers

Error Code	Description
9024	The username parameter is missing from the LDAP request control.

If the directory is able to compare verifiers, and the comparison evaluates as false, the directory sends the standard error LDAP_COMPARE_FALSE to the client. Similarly, if the user being authenticated lacks a directory entry, the directory sends the standard error LDAP_NO_SUCH_OBJECT.

> **See Also:**
>
> Password Verifier Schema Elements in *Oracle Fusion Middleware Reference for Oracle Identity Management*.

A.16.1.15 Troubleshooting Oracle Internet Directory Password Wallets

The Oracle Internet Directory Server has two password wallets: `oidpwdlldap1` and `oidpwdr`*`SID`*.

The `oidpwdlldap1` file contains the DN and password of an ODS user in encrypted format. The Oracle Internet Directory server uses the credential to connect to the back end database at startup time.

- Oracle Internet Directory Server Does Not Start
- Password Not Synchronized

A.16.1.15.1 Oracle Internet Directory Server Does Not Start

The `oidctl` daemon process fails to start an Oracle Internet Directory server instance.

Problem

The password stored in the `oidpwdlldap1` wallet is not synchronized with the ODS password in the back end database.

Solution

Try to connect to the database again using the `sqlplus` command:

```
sqlplus ods /ods_password@connect_string
```

If the connection succeeds, try to synchronize the password in the wallet with the ODS password by using the `oidpasswd` tool to create a new wallet with the correct password. For example, ensure that `DOMAIN_HOME` is set, then type:

```
>> oidpasswd connect=connect_string create_wallet=true
```

If the connection attempt fails, you must login into the back end database as a database administrator and change the ODS password by using the sql command:

```
>> alter user ods identified by some_new_password
```

Then try to create a new `oidpwdlldap1` to store the new password.

Solution

Try to start the Oracle Internet Directory server again.

The `oidpwdr`*`SID`* file contains the DN and password of a replica DN in an encrypted format. The Oracle Internet Directory replication server uses the credential to connect to the Oracle Internet Directory server at startup time.

This is an example of a replication password wallet, `oidpwdr`*`SID`*:

```
/------BEGIN REPL CREDENTIAL:cn=replication dn,orclreplicaid=qdinh-sun_
adeldap,cn=replication configuration-----
ezNkZXMtY2JjLXBrY3M1cGFkfQUnaz0TsfzcP0nM1HcHAXchf5mJw+sb4y0bLvvw3RvSg7H
S7/WsKJB02fdSGRlmfWAV+6llkRQ26g==
-----END REPL CREDENTIAL:cn=replication dn,orclreplicaid=qdinh-sun_
adeldap,cn=replication configuration-----/
```

A.16.1.15.2 Password Not Synchronized

Either `oidctl` or `wlst` fails to start an Oracle Internet Directory server instance and the replication server log file `oidrepld00.log` reports that it is not able to bind.

Problem

The replica DN password stored in the `oidpwdr`*`SID`* is not synchronized with the replica DN password in the Oracle Internet Directory server.

Solution

Try to connect to the Oracle Internet Directory server instance using the `ldapbind` command. Specify the replica DN stored in `oidpwdr`*`SID`* and the replica DN password. For example:

```
>> ldapbind -h host -p port -D "cn=replication dn,orclreplicaid=qdinh-sun_adeldap,
cn=replication configuration" -q
```

If the connection succeeds, then you can reset the password in the `oidpwdr`*`SID`* wallet using `remtool` with the option `-pchgwalpwd`, which changes the password of the replication DN of a replica only in the wallet. If you do not remember the replication dn password, then you can reset it using `remtool` with the option `-prestpwd`, which resets the password of the replication dn of a replica.

After resetting the replication password wallet, restart the replication server instance again a using `start(name='`*`instance-name`*`',type='OID')` command.

A.16.1.16 Troubleshooting bulkload Errors

Oracle highly recommends that you investigate and correct all errors thrown by `bulkload` before proceeding with the next step. If you ignore an `bulkload` error, you are likely to run into serious problems later.

To get more information about the reason for error, run the command with debug enabled (`debug=t`). Debug information is available in *`$DOMAIN_HOME`*`/tools/OID/logs/bulkload.log` and in the database `ods.ds_ldap_log` table.

Most `bulkload` errors occur during data load or during index creation.

Problem

The `bulkload` command-line tool fails during data load.

Solution

Restore the directory to the state it was in before the data load by using one of these methods:

Solution

- Use the `bulkload recover` option
- Restore the database from a backup taken before you invoked `bulkload`.

Problem

The `bulkload` command-line tool fails during index creation.

Solution

Examine `bulkload.log`. Find and fix the specific issue that caused index creation failure. Run `bulkload` with the `index` option again.

Failure to correct index errors can cause duplicate entries or duplicate rows in the Oracle Internet Directory's tables.

Problem

The `bulkload` command-line tool fails because of a broken connection to the database. This can occur, for example, due to a host crash or in to a failover in Real Application Clusters.

Solution

Follow the following procedure:

1. Ensure that the database is restarted properly.
2. If the `bulkload` invocation employed only the `check="TRUE"` or `generate="TRUE"` options, but not the `load="TRUE"` option, go to step 3.

   If it was the `bulkload load="TRUE"` option that failed, you must restore the database to its state before the failure. How you do that depends on whether you have a backup of the database before you issued the `bulkload load="TRUE"` command.

   - If you have a backup, use it to restore the database to its original state before you issued the `bulkload` command.
   - If you do not have a backup, use the `bulkload recover` command to return the database to its state before the `bulkload load="TRUE"` command.
3. Re-issue the `bulkload` command that failed.

A.16.1.17 Troubleshooting bulkdelete, bulkmodify, and ldifwrite Errors

Oracle highly recommends that you investigate and correct all errors thrown by the bulk tools before proceeding with the next step. To get more information about the reason for error, run the command with debug enabled (`debug=t`).

Debug information is available in the corresponding log file, `bulkdelete.log`, `bulkmodify.log`, or `ldifwrite.log`, under *`$DOMAIN_HOME`*`/tools/OID/logs`. In the database, debug information is available in the `ods.ds_ldap_log` table.

Problem

The `bulkdelete` or `bulkmodify` command-line tool fails because of a broken connection to the database. This can occur, for example, due to a host crash or in to a failover in Real Application Clusters.

Solution

Ensure that the database is restarted properly. Then retry the `bulkdelete` or `bulkmodify` command that failed.

A.16.1.18 Troubleshooting catalog Errors

Oracle highly recommends that you investigate and correct all errors thrown by the bulk tools before proceeding with the next step. To get more information about the reason for error, run the command with debug enabled (`debug=t`).

.Debug information is available in *`$DOMAIN_HOME`*`/tools/OID/logs/catalog.log` and in database `ods.ds_ldap_log` table.

Problem

The `catalog` command-line tool fails because of a broken connection to the database. This can occur, for example, due to a host crash or in to a failover in Real Application Clusters.

Solution

Ensure that the database is restarted properly. Retry the `catalog` command that failed. If the original invocation employed the `add="TRUE"` option, the retry might fail because the first command partially completed. If the retry fails, use `catalog delete="TRUE"` to delete the attribute index, then retry the command again.

Problem

The `catalog` command throws an error because more than 1000 attributes are present in the file.

Solution

If you need to index more than 1000 attributes, use multiple files.

A.16.1.19 Troubleshooting remtool Errors

The `remtool` query may hang sometimes and the subsequent efforts to bind to the server with other tool might fail.

Problem

A remtool query such as

```
remtool -pdispqstat -v -bind host:port
```

hangs. While it is hanging, attempts to bind to the server with other tools might fail.

Solution

If there is a large backlog of changelogs waiting to be purged, the `remtool` search query runs for a long time. Ensure that changelog purging is configured appropriately for your environment. See Overview of Change Log Purging.

You can also increase the number of worker threads so that other tools can bind while remtool is running the query. See "Attributes of the Instance-Specific Configuration Entry" and the Oracle Internet Directory chapter in Oracle Internet Directory Performance Tuning in *Oracle Fusion Middleware Performance and Tuning Guide*.

A.16.1.20 Troubleshooting Server Chaining Error

This section provides information on troubleshooting server chaining error.

Problem

The log contains the error message `Server Chaining error` followed by `javax.naming.AuthenticationException`.

Solution

In ODSM, go to the **Advanced** tab and expand **Server Chaining**. In each enabled entry, click **Verify Login Credential**, **Verify User Container**, and **Verify Group Container**.

If the verification fails, examine the values you entered for errors. If the problem persists, consult the external directory administrator to verify the accuracy of the values you entered.

A.16.1.21 View Version Information

On the Oracle Directory Services Manager home page for Oracle Internet Directory, you can view version information about Oracle Directory Services Manager, Oracle Internet Directory, and the associated Oracle Database.

For information about using Oracle Directory Services Manager, see Overview of Oracle Directory Services Manager .

A.16.1.22 Troubleshooting Oracle Enterprise Manager Fusion Middleware Control and WLST

Oracle Enterprise Manager Fusion Middleware Control and WLST do not work after the system is patched to 11*g* Release 1 (11.1.1.4.0).

Problem

Oracle Enterprise Manager Fusion Middleware Control and WLST do not work after the system is patched to 11*g* Release 1 (11.1.1.4.0).

Solution

This problem occurs if you had SSL server authentication enabled and cipher suites configured prior to patching. To fix this problem after patching, remove the `orclsslciphersuite` attribute from the instance-specific configuration entry by using `ldapmodify`. The LDIF file for deleting the `orclsslciphersuite` attribute in the instance-specific entry is:

```
dn: cn=componentname,cn=osdldapd,cn=subconfigsubentry
changetype: modify
delete: orclsslciphersuite
```

The command is:

```
ldapmodify -D cn=orcladmin -q -p portNum -h hostname -f ldifFile
```

Restart Oracle Internet Directory.

Problem

Oracle Internet Directory is up and running, but you cannot change Oracle Internet Directory parameters by using Oracle Enterprise Manager Fusion Middleware Control or WLST. You might see the error message: `Unable to connect backend OID`.

Solution

This can occur if the Oracle Internet Directory port number was changed and the server was not restarted or the Oracle Internet Directory component registration was not updated.

Solution

This occurs if you specify an SSL port configured for server authentication or mutual authentication when using the replication wizard. The replication wizard can only connect to SSL ports that are configured for no authentication. Always specify a non-SSL port or an SSL port configured for no authentication when prompted to log in or when specifying a node.

A.16.1.23 Troubleshooting Oracle Directory Services Manager

This section lists issues related to Oracle Directory Services Manager.

This section includes the following topics:

- Cannot Invoke ODSM from Fusion Middleware Control
- Invoking ODSM from Fusion Middleware Control in Multiple NIC and DHCP Enabled Environment Fails
- Resolving Failover Issues
- ODSM Displays an Error Message
- Cursor Loses Focus
- Second popup of ODSM displays an Unresolvable Error

A.16.1.23.1 Cannot Invoke ODSM from Fusion Middleware Control

Problem

You attempt to invoke Oracle Directory Services Manager from Oracle Enterprise Manager Fusion Middleware Control by selecting **Directory Services Manager** from the Oracle Internet Directory menu in the Oracle Internet Directory target, then **Data Browser**, **Schema**, **Security**, or **Advanced**.

ODSM does not open. You might see an error message.

Solution

This is probably an installation problem. See OID with ODSM and Fusion Middleware Control in a New WebLogic Domain in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*

A.16.1.23.2 Invoking ODSM from Fusion Middleware Control in Multiple NIC and DHCP Enabled Environment Fails

Problem

The WebLogic Managed Server where Oracle Directory Services Manager is deployed has multiple Network Interface Cards (NIC) or is DHCP enabled. Attempts to invoke Oracle Directory Services Manager from Oracle Enterprise Manager Fusion Middleware Control fail and return `404 errors`.

Solution

Use the WebLogic Server Administration Console to change the listen address of the Managed WebLogic Server so that the IP address or hostname in the URL for Oracle Directory Services Manager is accessible.

Perform the following steps:

1. Using a web browser, access the WebLogic Server Administration Console.
2. In the left pane of the WebLogic Server Administration Console, click **Lock & Edit** to edit the server configuration.
3. In the left pane of the WebLogic Server Administration Console, expand **Environment** and select **Servers**.
4. On the Summary of Servers page, click the link for the WebLogic Managed Server where Oracle Directory Services Manager is deployed.
5. On the Settings page for the WebLogic Managed Server, update the Listen Address to the host name of the server where Oracle Directory Services Manager is deployed.
6. Click **Save** to save the configuration.
7. Click **Activate Changes** to update the server configuration.

A.16.1.23.3 Resolving Failover Issues

Problem

When you perform an Oracle Directory Services Manager failover using Oracle HTTP Server, the failover is not transparent.

Solution

The problem sequence and its resolution are as follows:

1. Oracle Directory Services Manager is deployed in a High Availability active-active configuration using Oracle HTTP Server.
2. Display an Oracle Directory Services Manager page using the Oracle HTTP Server name and port number.
3. Make a connection to an Oracle Internet Directory server.
4. Work with the Oracle Internet Directory server using the current Oracle Directory Services Manager Oracle HTTP Server host and port.
5. Shut down one managed server at a time using the WebLogic Server Administration Console.
6. Go back to the Oracle Directory Services Manager page and port, and the connection which was established earlier with Oracle Internet Directory. When you do, a message is displayed advising you to re-establish a new connection to the Oracle Directory Services Manager page.
7. If you encounter this problem, in your web browser, exit the current Oracle Directory Services Manager page.
8. Launch a new web browser page and specify the same Oracle Directory Services Manager Oracle HTTP Server name and port.
9. Re-establish a new connection to the Oracle Internet Directory server you were working with earlier.

> **Note:**
>
> - Oracle Directory Services Manager High Availability in *The Oracle Fusion Middleware High Availability Guide*for more information about Oracle Directory Services Manager in High Availability configurations.
> - "Configuring Oracle HTTP Server to Support Oracle Directory Services Manager in an Oracle WebLogic Server Cluster"

Problem

ODSM temporarily loses its connection to Oracle Internet Directory and displays the message `LDAP Server is down`.

Solution

In a High Availability configuration where ODSM is connected to Oracle Internet Directory through a load balancer, ODSM reports that the server is down during failover from one instance of Oracle Internet Directory to another. In other configurations, this message might indicate that Oracle Internet Directory has been shut down and restarted. In either case, the connection is reestablished in less than a minute, and you are able to continue without logging in again.

Problem

ODSM temporarily loses its connection to an Oracle Internet Directory instance that is using an Oracle RAC database. ODSM might display the message `Failure accessing Oracle database (oracle errcode=`*`errcode`*`)`, where *`errcode`* is one of the following values: `3113`, `3114`, `1092`, `28`, `1041`, or `1012`.

Solution

This error can occur during failover of the Oracle Database that the Oracle Internet Directory instance is using. The connection is reestablished in less than a minute, and you are able to continue without logging in again.

A.16.1.23.4 ODSM Displays an Error Message

Problem

ODSM displays the error message: `Error :Posn: -1, Size: 0`

Solution

This error can be ignored. It usually indicates that Oracle Internet Directory has detected an error in an ODSM operation. JNDI, which ODSM uses to connect to Oracle Internet Directory, sometimes returns this error code instead of the actual error code. Oracle Internet Directory server log files show a more meaningful error message.

A.16.1.23.5 Cursor Loses Focus

Problem

When you access ODSM in accessibility mode, using only the keyboard, in Internet Explorer 7, the cursor loses focus. This behavior has been observed under the following circumstances:

- You access the directory in SSL-enabled mode and the server certificate appears.

- You type an invalid password and the error dialog appears.

Solution

Press the Tab key nine times, then press the Enter key.

A.16.1.23.6 Second popup of ODSM displays an Unresolvable Error

Problem

You can invoke Oracle Directory Services Manager from Oracle Enterprise Manager Fusion Middleware Control by selecting **Directory Services Manager** from the Oracle Internet Directory menu in the Oracle Internet Directory target, then **Data Browser**, **Schema**, **Security**, or **Advanced**.

A new browser window, containing the ODSM Welcome screen, pops up. For example, if select the Schema tab, a popup window opens up and the Schema page gets loaded.

Now, if you attempt to invoke a second ODSM from Oracle Enterprise Manager Fusion Middleware Control by selecting **Directory Services Manager** from the Oracle Internet Directory menu in the Oracle Internet Directory target, then **Data Browser**, **Schema**, **Security**, or **Advanced**.

A second browser window pops up and ODSM displays the following error: `An unresolvable error has occurred. Please contact your administrator for more information.`

Solution

When the error screen comes up, click on the browser back button, and it will take you back to the ODSM page.

A.16.1.24 Troubleshooting a Locked User Account

A user account can sometimes become locked because of multiple `ldapbind`, `ldapcompare`, or `ldapsearch` operations performed by a user using the wrong credentials (password). Applications that depend on this user account can then fail to operate correctly until the account is unlocked.

To unlock the user account, you must first determine the unknown source (IP address) providing the wrong credentials for the LDAP operations.

To find the unknown source causing this problem, follow these steps:

1. Set the debug level to enable logging to capture all incoming `ldapbind`, `ldapcompare`, and `ldapsearch` operations.

   The following examples use `oid1` as the component name. The generated log files will be in the following directory:

   `$DOMAIN_HOME/servers/OID/logs/`*`componentName`*

   a. Create a file named `debug.ldif` with the following content:

   ```
   dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
   changetype: modify
   replace: orcldebugflag
   orcldebugflag: 1

   dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
   changetype: modify
   ```

```
replace: orcldebugop
orcldebugop: 193
```

The `debug.ldif` file enables debugging by setting `orcldebugflag` to 1. The file also sets the `orcldebugop` flag to 193 to log `ldapbind`, `ldapcompare`, and `ldapsearch` operations, which are the basic operations that can cause a user account to get locked.

For information about setting the `orcldebugop` flag, see Table 24-4.

b. Load the LDIF file using the `ldapmodify` command. For example:

```
ldapmodify -h oid_host -p oid_port -D cn=orcladmin -p admin_password -f
debug.ldif
```

For more information about logging, see Managing Logging.

2. After waiting for the problem to reoccur, check for information about the `ldapbind`, `ldapcompare`, and `ldapsearch` operations in the following Oracle Internet Directory server logs:

```
$DOMAIN_HOME/servers/OID/logs/componentName/oidldap01s*.log
```

In this log location, `oidldap01s*.log` specifies all server logs for the `oid1` component. For more information, see Table 24-1.

Because these logs will contain a large amount of information, find the relevant information by searching through the file for `"useraccount"`. On some systems, you can use a `grep` command. For example:

```
cd $DOMAIN_HOME/servers/OID/logs/componentName
grep -A 5 -B 3 "useraccount" oidldap01s*.log
```

In this example, `"useraccount"` is the user account that is locked. The -A and -B options, which are supported on some systems, return five lines above and three lines below the line with the user account.

For example, if `faadmin` is the user account that is locked, the following `grep` command returns the relevant lines:

```
grep -A 5 -B 3 "faadmin" oidldap01s*.log
```

3. Analyze the output of the `grep` command. For example, consider this sample output:

```
idldapd01s20149-0082.log-[2012-06-26T15:54:45-07:00] [OID] [TRACE:16] []
[OIDLDAPD] [host: adcdk04] [pid: 20149] [tid: 13] [ecid:
004kqVw6GZG2nJK_ITDCif0004up00Nd8Q,0] ServerWorker (REG):[[
oidldapd01s20149-0082.log-BEGIN
oidldapd01s20149-0082.log ConnID:40614 mesgID:306 OpID:25 OpName:bind
ConnIP:::ffff:10.240.109.21
ConnDN:cn=faadmin,cn=users,dc=us,dc=example,dc=com
oidldapd01s20149-0082.log:INFO : gslfbidbDoBind * Version=3 BIND
dn="cn=FAAdmin,cn=Users,dc=us,dc=example,dc=com" method=128
oidldapd01s20149-0082.log- ConnId = 40614, op=25, IpAddr=::ffff:10.240.109.21
oidldapd01s20149-0082.log-2012-06-26T15:54:45 * Adding pwdfailuretime in
gslsbmApplyModtoEntry()
oidldapd01s20149-0082.log-2012-06-26T15:54:45 * INFO:gsleswrASndResult
OPtime=4929 micro sec RESULT=49 tag=97 nentries=0
oidldapd01s20149-0082.log-END
oidldapd01s20149-0082.log-]]
```

This output indicates the IP address of the source that is executing the LDAP operations using the wrong credentials:

- Request ID (remains constant for a particular request): ecid: 004kqVw6GZG2nJK_ITDCif0004up00Nd8Q,0
- IP Address: IpAddr=::ffff:10.240.109.21
- Operation Name: OpName:bind
- Bind DN: BIND dn="cn=FAAdmin,cn=Users,dc=us,dc=example,dc=com"

4. Using information from the previous step, stop the operation causing the problem.

> **Note:**
>
> Multiple LDAP operations might be causing the user to be locked, so continue searching the log file for the source of any other operations that should be stopped.

5. After you have stopped the operations causing the problem, you can disable debugging by setting the `orcldebugflag` and `orcldebugop` values to 0:

   a. Create a file named `debug_disable.ldif` with the following content:

```
dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orcldebugflag
orcldebugflag: 0

dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
changetype: modify
replace: orcldebugop
orcldebugop: 0
```

   b. Load the LDIF file using the `ldapmodify` command. For example:

```
ldapmodify -h oid_host -p oid_port -D cn=orcladmin -p admin_password -f
debug_disable.ldif
```

A.16.1.25 Troubleshooting Policy Store Migration

This section describes the procedure to troubleshoot policy store migration related errors.

If Oracle Internet Directory is used as the Policy Store, during the policy migration from a Fusion Applications dedicated environment to a shared environment, migration of the Security Store can run very slowly because of slow OPSS queries to Oracle Internet Directory. In some cases, the migration can fail because of a timeout.

Solution

To improve OPSS query performance, set the following tuning values for the Oracle Internet Directory Policy Store:

- **Oracle Database Tuning Parameters**
  - `SGA_MAX_SIZE`: 4G or higher
  - Oracle Database server processes: 500 or higher
- **Oracle Internet Directory Attributes**
  - `orclecacheenabled`: 2 (Enable both Entry Cache and Result Set Cache.)
  - `orclrscacheattr` - Set multi-valued attribute as follows:

```
orclrscacheattr: orcljaznprincipal
orclrscacheattr: orcljaznpermissiontarget
orclrscacheattr: orcljpsresourcename
orclrscacheattr: uniquemember
orclrscacheattr: orcljpsassignee
```

- `orclecachemaxsize`: 16G or higher
- `orclinmemfiltprocess` - Set multi-valued attribute as follows:

```
orclinmemfiltprocess: (orcljpsresourcetypename=taskflowresourcetype)
orclinmemfiltprocess: (orcljpsresourcetypename=regionresourcetype
```

A.16.1.25.1 Troubleshooting Policy Store Migration With Oracle ZFS

If you are using Oracle ZFS and are experiencing performance issues during Policy Store migration, the issue might be related to the Oracle Database using Dynamic Intimate Shared Memory (DISM). The ZFS file system and DISM might be locking over memory page access.

Solution

Set the `SGA_MAX_SIZE` and `SGA_TARGET` to the same size (4G), which effectively disables DISM in the database.

A.16.2 Need More Help?

If you could not find the troubleshooting information you were looking for, then visit the following links:

You can find additional solutions to problems at these sites:

- My Oracle Support (formerly MetaLink): `https://support.oracle.com`
- *Oracle Fusion Middleware Release Notes* collection, available on the Oracle Technology Network (OTN):

  `http://www.oracle.com/technology/documentation`

If you do not find a solution for your problem, log a Service Request with Oracle.

To help Oracle Support Services troubleshoot your problem, provide the following information:

- A detailed description of how and when the problem occurred, including:
  - The commands, procedures, or operations that might have triggered the problem.
  - Whether the problem is reproducible.
  - Whether the problem is caused by a standalone, reproducible LDAP operation that can be invoked using a command such as `ldapsearch`, `ldapadd`, `ldapmodify`, or `ldapdelete`.
- Oracle Internet Directory debug logs, as described in the next section.

A.16.2.1 Oracle Internet Directory Debug Logs

Oracle Internet Directory debug logs can be helpful in finding a solution to a problem.

> **Note:**
>
> - Alert logging capability helps in logging poorly performing requests in separate alert log files. This helps in tracking outliers so that such information is not lost in the regular diagnostic logs. Schema cache refresh is also recorded as part of these alert log files.
> - Alert log files contain essential information about database performance metrics. The DB logging is done irrespective of log levels for all operations if operation's total time is greater than `orclmaxlatencylog` defined on DSA configuration entry. See Attributes of the DSA Configuration Entry for more information on the attributes.
> - The individual SQLs that are executed as part of the request processing and the corresponding time that is spent in the database will be recorded in regular OID server diagnostic files.

The debug logs are generated in the following directory:

`$DOMAIN_HOME/servers/OID/logs/`*`componentName`*

Here *componentName* is the Oracle Internet Directory instance component name. Examples in this section use `oid1` as the *componentName*.

To generate Oracle Internet Directory debug logs for a specific problem:

1. Set the debug logging level to capture information about your specific problem, as described in Setting Debug Logging Levels Using the Command Line.

   The `orcldebugflag` attribute determines the debug logging levels for Oracle Internet Directory.

   For example, if you are troubleshooting Access Control List (ACL) related problems, LDAP errors such as access denied (LDAP error code 49) are displayed when users, roles, or attributes are added to Oracle Internet Directory.

   In this situation, debugging information for ACL processing and heavy trace debugging is needed in order to obtain a view of the ACLs at each level in the DIT.

   The value for heavy trace debugging is 1 and the value for ACL processing is 8192. Therefore, you would set `orcldebugflag` to 8193 (1 + 8192).

   For a list of all values you can specify, see Table 24-3.
2. After you set the debug level, either perform the operation again that might have triggered the problem or wait until the problem occurs again.
3. After the problem occurs, check the debug logs in the following directory:

   `$DOMAIN_HOME/servers/OID/logs/oid1/oidldap01s*.log`

   In this log location, `oidldap01s*.log` specifies the server logs for the `oid1` component. For more information, see Table 24-1.

4. Because the debug logs can be very large, do not send entire log files to Oracle. Search the logs to find the lines that describe the error and then provide a snippet from the file to help Oracle with the troubleshooting process.

   You can often identify the location of these lines in the log file based on the time when the operation that triggered the issue was performed.