Oracle® Communications Calendar Server

Installation and Configuration Guide

Release 8.0
E63135-04
June 2025

Oracle Communications Calendar Server Installation and Configuration Guide, Release 8.0

E63135-04

Contents

Preface

1 Calendar Server Installation Overview

2 Planning Your Calendar Server Installation

3 Calendar Server System Requirements

4 Calendar Server Pre-Installation Tasks

5 Installing Calendar Server

6 Configuring Calendar Server with Multiple Hosts

7 Configuring the Calendar Server Document Store

12 Uninstalling Calendar Server

13 Installing Patches

A commpkg Reference

B Calendar Server Configuration Scripts

C comm_dssetup.pl Reference

D rundssetup Reference

E ODSEE to OUD Migration

Preface

This guide provides instructions for installing and configuring Oracle Communications Calendar Server.

Audience

This document is intended for system administrators or software technicians who install and configure Calendar Server. This guide assumes you are familiar with the following topics:

- MySQL Server or Oracle Database
- Oracle GlassFish Server or Oracle WebLogic Server
- Oracle Directory Server Enterprise Edition and Oracle Unified Directory Server
- System administration and networking

Nomenclature

The following nomenclature is used throughout the document.

Convention	Meaning
Application Server	The term Application Server or application server is used in this document to refer to either GlassFish Server or WebLogic Server. Supported Application Server: Oracle Communications Calendar Server 8.0.0.3.0 and previous releases were deployed on GlassFish Server, which is no longer supported by Oracle. For that reason, Calendar Server 8.0.0.4.0 and beyond are only supported on Oracle WebLogic Server. Oracle strongly recommends that you upgrade your Calendar Server environments to release 8.0.0.4.0 or higher and migrate to WebLogic Server to receive full Oracle support.
Directory Server	The term Directory Server or directory server is used in this document to refer to either Oracle Unified Directory Server (OUD) or Oracle Directory Server Enterprise Edition (ODSEE). Oracle strongly recommends that you migrate from ODSEE to OUD to receive full Oracle support.

> **Note:**
>
> Calendar Server release 8.0.0.6.0 and above support Oracle Unified Directory Server 12.2.1.4. For Oracle Unified Directory support, you need to install and run DS setup 6.4.0.30.0. For more information, see the appendices "rundssetup Reference" and "ODSEE to OUD Migration". ODSEE is not supported from 8.0.0.8.20250424. Calendar Server 8.0.0.8.20250424 supports MySQL Server 8.4. Calendar Server 8.0.0.6.0 and above support MySQL Server 8.0 in addition to MySQL Server 5.7.x. Calendar Server 8.0.0.6.0 and above does not support MySQL Server 5.5.x.

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.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1 Calendar Server Installation Overview

This chapter provides an overview of the Oracle Communications Calendar Server installation process.

Overview of Calendar Server Installed Components

During the installation process, you install and configure the following components:

- Java
- Application server
- Either MySQL Server or Oracle Database
- Calendar Server

Calendar Server depends on Directory Server for LDAP services.

If your site does not currently have Directory Server deployed, and you need to install either Oracle Directory Server Enterprise Edition or Oracle Unified Directory (depending on the Calendar version you are using). For more information, see:

- ODSEE: ODSEE Documentation
- OUD: OUD Documentation

To configure Oracle Unified Directory (OUD), use the **rundssetup** script.

Before running DS Setup, ensure that a Directory Server instance is already created, and install the following:

- OUD 12.2.1.4
- Python 2.7 and above

> **Note:**
>
> Calendar Server release 8.0.0.6.0 and above support Oracle Unified Directory Server 12.2.1.4. ODSEE is not supportd from 8.0.0.8.20250424. For Calendar Server to use notifications, you must also have an email server installed, such as Oracle Communications Messaging Server.

Overview of the Calendar Server Installation Procedure

The installation procedure follows these steps:

1. Plan your installation. When planning your installation, do the following:
   - Determine the scale of your implementation, for example, a small development system, or a large production system.

- Determine how many physical machines you need, and which software components to install on each machine.
- Plan the system topology, for example, how the system components connect to each other over the network.

2. Review system requirements. System requirements include:
   - Hardware requirements, such as disk space.
   - System software requirements, such as operating system (OS) versions and OS patch requirements.
   - Information requirements, such as IP addresses and host names.
3. Install and configure software upon which Calendar Server is dependent, including:
   - Java
   - Application server
   - Either MySQL Server or Oracle Database
4. Prepare the Directory Server schema by installing and running either **comm_dssetup.pl** or **rundssetup** (depending on the Calendar version you have) script from the Calendar Server media pack.

   > **Note:**
   >
   > DS Setup version 6.4.0.30.0 uses **rundssetup** script which is an enhanced version of **comm_dssetup.pl** script available in earlier versions of DS Setup. Calendar Server release 8.0.0.6.0 and above support Oracle Unified Directory Server 12.2.1.4. Oracle recommends to use OUD and **rundssetup**.

   For more information, see "rundssetup Reference".
5. Install and configure Calendar Server.
6. (Optional) Configure additional Calendar Server front ends and back ends for a multiple host deployment.
7. Configure the document stores.
8. Perform post-installation configuration tasks.
9. Verify the installation.

After Calendar Server is installed, you might perform additional security-related tasks. For more information, see *Calendar Server Security Guide*.

Calendar Server Installation Options

You install Calendar Server in either interactive or silent mode. When you run the installer in silent mode, you are running a non-interactive session. The installation inputs are taken from the following sources:

- A silent installation file
- Command-line arguments
- Default settings

You can use silent mode to install multiple instances of the same software component and configuration without having to manually run an interactive installation for each instance.

Ensuring a Successful Calendar Server Installation

Only qualified personnel should install the product. You must be familiar with the UNIX operating system and the application server. You should be experienced with installing Java-related packages. Oracle recommends that an experienced database administrator install and configure database software.

Follow these guidelines:

- As you install each component, for example, the Oracle database and Application Server, verify that the component installed successfully before continuing the installation process.
- Pay close attention to the system requirements. Before you begin installing the software, make sure your system has the required base software. In addition, ensure that you know all of the required configuration values, such as host names and port numbers.
- As you create new configuration values, write them down. In some cases, you might need to re-enter configuration values later in the procedure.

Directory Placeholders Used in This Guide

Table 1-1 lists the placeholders that are used in this guide:

Table 1-1 Calendar Server Directory Placeholders

Placeholder	Directory
CalendarServer_home	Specifies the installation location for the Calendar Server software. The default is **/opt/sun/comms/davserver**.
InstallRoot	Specifies the installation location for the Unified Communications Suite software. The default is **/opt/sun/comms**.
GlassFish_home	Specifies the installation location for the Oracle GlassFish Server software. The default is **/opt/glassfish3/glassfish**.
WebLogic_home	The directory in which Oracle WebLogic Server software is installed.
GlassFish_Domain	Oracle GlassFish Server domain in which Calendar Server is deployed. For example: *GlassFish_home/domains/domain1*
Weblogic_Domain	Oracle WebLogic Server domain in which Calendar Server is deployed. For example, *WebLogic_home/user_projects/domains/base_domain* **Note:** In case of WebLogic Server, it must be configured with at least one Managed Server instance and the Managed Server instance must host Calendar Server.
AppServer_Domain	The domain of the application server in which Calendar Server is deployed.

2

Planning Your Calendar Server Installation

This chapter provides information about planning your Oracle Communications Calendar Server installation. It also describes the Calendar Server logical and physical architectures.

About Calendar Server

Calendar Server is a high-performance, Internet standards-based calendar server that features calendaring, group scheduling, event-information sharing, task management, event and task search, and email invitations. Calendar Server supports the standard CalDAV access protocol, which makes it usable with Apple iCal, iPhone, Thunderbird Lightning, and any other CalDAV client.

Calendar Server Front-End and Back-End Components

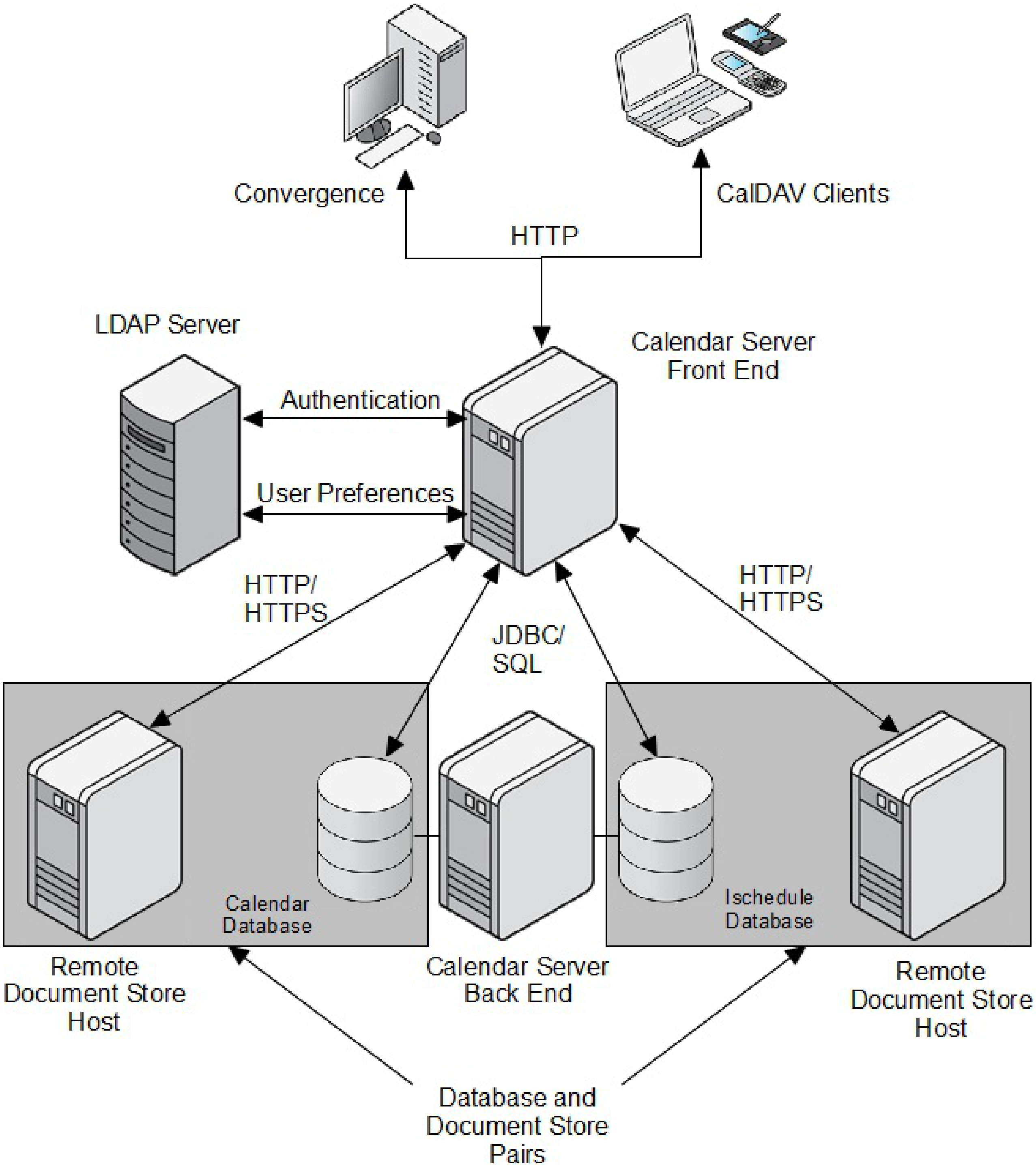
Calendar Server consists of the following front-end and back-end components:

- Stateless J2EE-based front end (provided by the application server)
- SQL back end (can be either MySQL Server or Oracle Database), for calendar data and iSchedule data
- Document store, for storing large data

You can locate these components on the same host or separate the components onto multiple hosts.

Figure 2-1 shows a Calendar Server configuration that uses one front-end host and one back-end host. In this figure, the back-end host contains both the calendar database and the iSchedule database. The document stores, one for each of the databases, are located on separate hosts.

Figure 2-1 Calendar Server Front-End and Back-End Configuration



In a multiple-host deployment, each front-end host has access to all the SQL back-end hosts. As a consequence, each front-end host has access to all Calendar Server end users' data.

Multiple front ends can be grouped together by using a simple load balancer. The load balancer must use IP-based stickiness as its load balancing method.

Mapping Calendar Users to Back-End Servers

In a multiple back-end host environment, when a user issues a Calendar Server request, the front-end host handling the request must determine on which back-end SQL host the targeted resource resides. Calendar Server makes this determination by using the "davstore" identifier (the actual LDAP attribute is **davStore**).

The mapping between Calendar Server front ends and back ends is indirect:

- Each Calendar Server front-end instance is configured with a list of opaque back-end identifiers, called "davstore" identifiers. There is one "davstore" identifier per back-end SQL host database.
- A Java Naming and Directory Interface (JNDI) name is associated to each "davstore" identifier (it is also in the Calendar Server configuration), for example: **jdbc/***davstore_id*.
- The application server that hosts the Calendar Server is configured with corresponding JNDI resources containing the actual back-end information such as server names, ports, database names, credentials, and so on.
- At startup, Calendar Server performs a JNDI lookup for each "davstore" identifier value in its configuration.

davStore LDAP Attribute

You can add the **davStore** LDAP attribute to users' and resources' object classes to associate those users and resources with a particular back-end Calendar Server store. The value of the **davStore** attribute is equal to one of the davstore IDs defined in the server configuration. The **davStore** LDAP attribute is single valued. When not present, a server configurable default davstore ID is used.

Users have only one **davStore** LDAP attribute for all types of data. In the single user repository model, all data for users with the same **davStore** value is hosted on the same back-end repository. This does not prevent a deployment from using different repositories for the different types of data.

Planning Your Calendar Server Installation

This section contains the following planning topics you must consider before installing Calendar Server:

- Using a Single Web Container for Multiple Applications
- Deciding on the Database
- Planning for Multiple Calendar Server Hosts
- Planning the Document Store
- Planning the iSchedule Database
- Planning for Virus Scanning
- Planning to Use an External Directory for Authentication
- Planning for the Unique Identifier

Using a Single Web Container for Multiple Applications

You use the application server as a web container for Calendar Server. For production deployments, deploy Calendar Server to the root context (/) to simplify configuring mobile clients. The URI syntax is:

```
http://calendar_server_host_name:port/calendar_server_webapp/dav/principals/email-address/
```

For example:

```
http://example.com:3080/davserver/dav/principals/jsmith@example.com/
```

You can deploy Calendar Server with other applications in the same GlassFish Server web container. In this case, you deploy Calendar Server in its own context, for example, **/davserver**, and not the root context (/) of the GlassFish instance.

If you use Oracle WebLogic Server as a web container, ensure to use a separate Managed Server for deploying Calendar Server.

Regardless of whether you use a single web container for multiple applications or put Calendar Server in its own web container, for production environments, deploy Calendar Server under the root (/) URI context.

Deciding on the Database

You can use either MySQL Server or Oracle Database as the Calendar Server database to store calendar and iSchedule information. You cannot mix database types in a deployment.

> **Caution:**
>
> You can view contents of the database by using standard SQL tools. However, do not use SQL tools to modify your data. This applies to both MySQL Server and Oracle Database.

Planning the iSchedule Database

The iSchedule database is used to manage external calendar invitations.

The established standard for scheduling between two separate calendar servers is still only through iCalendar Message-Based Interoperability Protocol (iMIP), which sends calendar data over email. Previously, end users had to manually import such invitations and responses that arrived in email into their calendars. Starting with Oracle Communications Messaging Server 7 Update 5, you can use an iSchedule channel that interprets such mail messages and posts them to Calendar Server directly. Thus, external invitations and responses get into users' calendars without any user intervention. See the topic about using the iSchedule channel to handle iMIP messages in *Messaging Server Administration Guide* for instructions on how to set up Messaging Server. No special setup is required for Calendar Server.

Planning the Document Store

The Calendar Server document store is used to store and retrieve *large* data, such as calendar events with many invitees, and todos with large attachments. You must set up one document store per configured Calendar Server back-end database. That is, every logically configured database must have its own unique document store to manage its large data. You must configure a document store even if you decide to not use its capability.

The document store can be either *local* or *remote*, or, if you use Oracle Database, within the database itself. When the document store is local, it runs as part of a Calendar Server instantiation. For example, the local document store could be part of a Calendar Server front-end installation or part of a single host installation providing both front-end and back-end functionality. When the document store is local, Calendar Server accesses the file systems directly. You can only use a local document store in a single front-end deployment, as all front-end hosts need to be able to access the data. When the document store is remote, Calendar Server accesses the documents by using either HTTP or HTTPS. If you use Oracle Database, you can configure it to contain both the calendar data and the data that otherwise would need to be stored in the separate document store. That is, you would use a single, *large* Oracle database for both calendar data and the document store.

> **Note:**
>
> You cannot configure MySQL database to contain both contact data and document store data.

In a multi-host deployment with multiple front-end hosts, you must ensure that each front-end host that accesses a back-end database must also have access to the back-end database's related document store.

The default directory for the document store is **/var/opt/sun/comms/davserver/db**. You can configure this by changing the **store.dav.*.dbdir parameter**. If at some point you move the document store, you must migrate the data.

Remote Document Store Authentication

Calendar Server provides enhanced security for remote document stores: a remote document store requires password authentication of the connection between the Calendar Server front-end host and the remote document store server.

Securing Connections to Remote Document Store

You can also configure secure communication between the Calendar Server front-end hosts and the remote document stores. See *Calendar Server Security Guide* for more information.

Planning for Multiple Calendar Server Hosts

Using multiple Calendar Server hosts can help you:

- Avoid network latency and unnecessary bandwidth consumption by positioning the server closer to the client (that is, in a geographically distributed environment).
- Scale your deployment by distributing end users onto different machines, thus avoiding possible bottlenecks in terms of I/O, memory, CPU, and backup time. A very large deployment can also be geographically distributed.

> **Note:**
>
> The Calendar Server default installation and configuration supports only one front-end/back-end deployment. You must perform some additional steps to set up the multiple-server scenario. See "Configuring Calendar Server with Multiple Hosts" for more information.

Planning for Virus Scanning

Calendar Server supports virus scanning of attachments, such as documents. If you choose to configure virus scanning, decide whether to use an existing Messaging Server MTA, or to deploy a dedicated MTA-only Messaging Server installation to scan for viruses. To use virus scanning for Calendar Server, you must deploy at least Messaging Server 7.4 patch 23. For more information, see the topic on configuring virus scanning in *Calendar Server System Administrator's Guide*.

Planning to Use an External Directory for Authentication

You can configure your deployment to authenticate against a separate, external LDAP directory while keeping user LDAP information internal and private. Such a configuration is useful in hosted environments for delegating one administrative aspect to a provider (managing the Calendar Server front- and back-end hosts) while maintaining control over the LDAP user passwords in the internal, corporate network. For more conceptual information on Calendar Server and external authentication, see *Calendar Server Concepts*. For instructions on how to configure external authentication, see to the topic on configuring external authentication in *Calendar Server System Administrator's Guide*.

Planning for the Unique Identifier

Calendar Server requires a unique identifier in the form of an LDAP attribute whose value is used to map each user account (in the LDAP Directory Server) to a unique account in the calendar database (the MySQL Server or Oracle Database) that stores the Calendar Server data. The unique identifier links various entries from different database tables for a user. You must use a unique identifier, and one that does not change, for each user entry stored in LDAP.

Before installing Calendar Server, decide on the LDAP attribute to be used as the unique identifier. This is a critical decision. It is very difficult to change the attribute you use as the unique identifier once you deploy Calendar Server and start using it. Calendar Server provides the **davuniqueid** attribute, which is the recommended attribute to use. For more information on this attribute, see *Communications Suite Schema Reference*.

How Do You Set the Unique Identifier?

You set the attribute to be used as the unique identifier during the Calendar Server initial configuration phase. (When installing a Communications Suite component such as Calendar Server, you first install the software and then configure it. These are two separate steps.) The Calendar Server configurator script, **init-config**, prompts you to enter the attribute you have chosen as your unique identifier. The configurator then stores the value to the Calendar Server **davcore.uriinfo.permanentuniqueid** configuration parameter.

In the initial release of Calendar Server 7, the LDAP operational attribute **nsUniqueId** was chosen as the default value used for the unique identifier. However, it was discovered that this choice has a potential serious downside. The problem with using **nsUniqueId** is that if the

LDAP entry for a user, group, or resource is deleted and recreated in LDAP, the new entry would receive a different **nsUniqueId** value from the Directory Server, causing a disconnect from the existing account in the calendar database. As a result, recreated users cannot access their existing calendars. See "Changing the User Unique Identifier" for more information.

To prevent this potentially serious issue, Calendar Server 7 Update 3 introduced a new attribute, **davUniqueId**, in the **davEntity** object class, to use as the unique identifier. You should provision this attribute for your users, groups, and resources LDAP entries with globally unique IDs. To use Delegated Administrator as your provisioning tool, ensure that you are running at least Delegated Administrator 7 Patch 6, which supports provisioning the **davUniqueId** attribute.

In the Calendar Server data base, the unique identifier value is case sensitive. If you must move or recreate the corresponding LDAP entry, make sure to retain the case of the value as is. However, because the value is considered as case insensitive for LDAP comparisons, do not create a unique identifier value for another user or resource entry by just changing the case of the value.

Setting the Unique Identifier During Calendar Server Installation

For a fresh Calendar Server installation use these steps to set the unique identifier:

1. During initial configuration, when prompted by the Calendar Server **init-config** configurator, choose the default value, **davUniqueId**.

   This is stored to the **davcore.uriinfo.permanentuniqueid** configuration parameter. For more information, see the topic on **davUniqueId** in *Communications Suite Schema Reference*.

2. In LDAP, populate calendar users, groups, and resources with the **davUniqueId** attribute.

   You can:

   - Use the **populate-davuniqueid** tool.
   - Use your own LDAP tools.

Changing the Unique Identifier During Calendar Server Upgrade

If you initially deployed Calendar Server 7 Update 2 to use the **nsUniqueId** attribute as your unique identifier, you should switch to using the **davUniqueId** attribute, new in Calendar Server 7 Update 3. For more information on the problems with using **nsUniqueId**, see "Changing the User Unique Identifier".

To upgrade and change to using the **davUniqueId** attribute, see "Upgrading Calendar Server".

System Deployment Planning

This section contains the following system-level planning topics you must consider before installing Calendar Server:

- Planning for High Availability
- Planning Backup Strategies

Planning for High Availability

You can configure Calendar Server front-end and back-end hosts to be highly available. For high availability of Calendar front-end hosts, deploy the hosts behind a load balancer. See

"Using Load Balancing" for more information. Choices for making the Calendar Server back-end database highly available include MySQL asynchronous replication and Oracle Data Guard. Refer to the documentation for those products for installation and configuration instructions. You can also configure the document store for high availability. See "Configuring the Document Store for High Availability" for more information.

Using Load Balancing

When you deploy multiple Calendar Server front-end hosts, a load balancer (with IP-based stickiness) is necessary to distribute the load across the front-end hosts.

Planning Backup Strategies

Backing up and restoring data is one of the most important administrative tasks for your Calendar Server deployment. You must implement a backup and restore policy for your Calendar Server database to ensure that data is not lost if the system crashes, hardware fails, or information is accidentally deleted.

The two ways to back up Calendar Server data are:

- **davadmin db backup** command
- Oracle Solaris ZFS snapshots

For more information, see *Calendar Server System Administrator's Guide*.

Calendar Server Logical Architecture

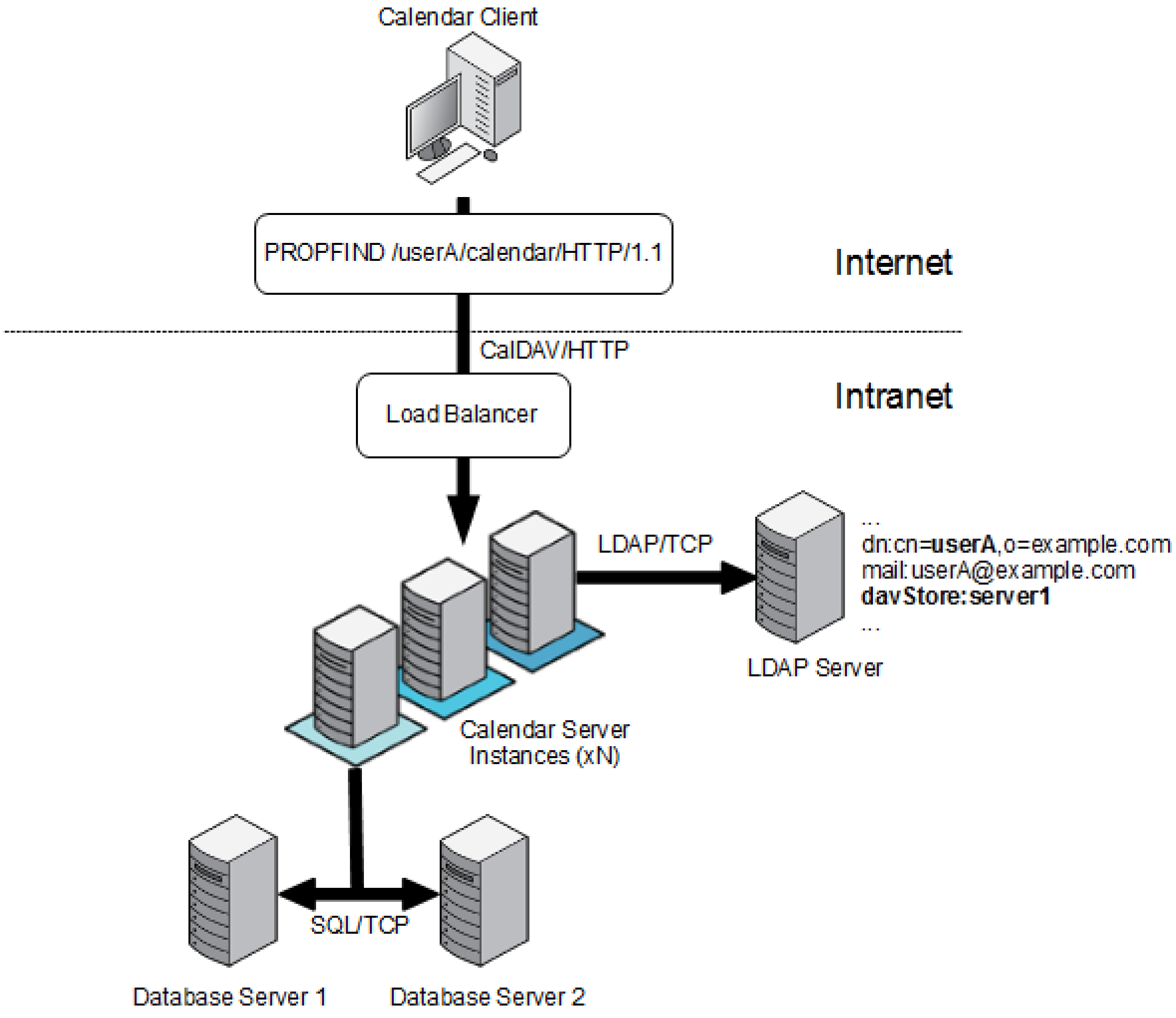
When planning your Calendar Server logical architecture, you can use the following options:

- **Single-tiered Calendar Server architecture:** You can deploy all components on a single host.
- **Two-tiered Calendar Server architecture:** You can deploy Calendar Server with the front-end component installed on a separate host and the database back end installed on another host.
- **Two-tiered, multiple server Calendar Server architecture:** You can install multiple front-end hosts and multiple back-end database hosts. You can also install the document store onto a separate remote host.

Sample Calendar Server Physical Architecture

Figure 2-2 shows a sample Calendar Server deployment consisting of three front-end hosts, two back-end hosts, and a load balancer to handle client requests.

Figure 2-2 Calendar Server Physical Architecture



About Installing a Secure System

You can configure Secure Sockets Layer (SSL) between the Calendar Server GlassFish Server front-end hosts and database back-end hosts. You can also configure SSL between the Calendar Server GlassFish Server front-end hosts and the remote document stores.

When configuring SSL between the front-end and back-end hosts, you first enable the back-end database hosts for SSL by configuring either the required **trustStore** files or wallets. Then you configure the Calendar Server front-end hosts to connect over SSL by making use of the stored certificates.

For information about secure installation and configuration of Calendar Server, see *Calendar Server Security Guide*.

If you use Oracle WebLogic Server, see the discussion about performing a secure Calendar Server installation chapter in *Calendar Server Security Guide*.

3

Calendar Server System Requirements

This chapter describes the hardware, operating system, software, and database requirements for installing Oracle Communications Calendar Server.

Software Requirements

Calendar Server is installed on the application server's domain. It uses either Oracle Database or MySQL Server for data storage.

Supported Operating Systems

Table 3-1 lists operating systems that support Calendar Server.

Table 3-1 Supported Operating Systems

Operating System	CPU	Required Patches
Oracle Solaris 11	SPARC, x64	See the Oracle Solaris documentation for patch information.
Oracle Linux 7, Oracle Linux 8 Note: Oracle Linux 8 is supported from Calendar Server 8.0.0.8.0.	ix64	See the Oracle Linux documentation for patch information.

Required Software

Table 3-2 lists the database software choices for Calendar Server.

Table 3-2 Database Requirements

Product	Version
Oracle Database	11g Release 2, 12c To upgrade to Oracle Database 12c from Oracle Database 11g, see *Oracle Database Upgrade Guide* at: `https://docs.oracle.com/database/121/UPGRD/toc.htm` **Note**: Calendar Server Patch 8.0.0.4.0 is certified with Oracle Database 19c.
MySQL Server	5.5.40, 5.6.22, 5.7.26, 8, and 8.4. To upgrade to MySQL Server 5.6 from MySQL Server 5.5, see "Upgrading from MySQL 5.5 to 5.6" at: `http://dev.mysql.com/doc/refman/5.6/en/upgrading-from-previous-series.html` Note: Calendar Server 8.0.0.6.0 and above support MySQL Server 8.0 in addition to MySQL Server 5.7.x. Calendar Server 8.0.0.6.0 and above does not support MySQL Server 5.5.x. Calendar Server 8.0.0.8.20250424 supports MySQL Server 8.4.

Note:

You cannot mix database types in a deployment.

Table 3-3 lists other software required for installing and running Calendar Server.

Table 3-3 Software Requirements

Product	Version	Notes
Directory Server	Oracle Directory Server Enterprise Edition 6.x, 7, 11.x or Oracle Unified Directory 12.2.1.4.0	For a fresh installation, use the latest version of Directory Server.
Application Server	GlassFish Server 3.1.2.x WebLogic Server 12.2.1.3.0 and 12.2.1.4.0.	Required as a web container: For GlassFish Server, download the patch from My Oracle Support at: `https://support.oracle.com` For WebLogic Server, download the generic installer from `http://edelivery.oracle.com` For WebLogic Server 12.2.1.4.0, the installer is available as **Oracle Fusion Middleware 12c (12.2.1.4.0) WebLogic Server and Coherence** and contains **fmw_12.2.1.4.0_wls.jar** and **fmw_12214_readme.html**. For example, for Linux Platform, you should download the **V983364-01.zip** package. **Supported Application Server**: Oracle Communications Calendar Server 8.0.0.3.0 and previous releases were deployed on GlassFish Server, which is no longer supported by Oracle. For that reason, Calendar Server 8.0.0.4.0 and beyond are only supported on Oracle WebLogic Server. Oracle strongly recommends that you upgrade your Calendar Server environments to release 8.0.0.4.0 or higher and migrate to WebLogic Server to receive full Oracle support.
Java	For GlassFish Server, use Java 7 For WebLogic Server, use Java 8	The Java version is according to the application server that you have selected to deploy Calendar Server. **Note**: You must download the latest security patch updates available for the respective Java version.

Client Requirements

Calendar Server supports any standard CalDAV client. Table 3-4 shows which clients were tested with Calendar Server.

Table 3-4 Calendar Server Clients

Client	Minimum Version	Notes
Convergence	2 (Patch 6), testing done with Convergence 3	Calendar Server Patch 8.0.0.8.20250424 is certified with Convergence 3.0.3.4.20241203.
Connector for Microsoft Outlook	8	NA
Apple iCal	6	iCal 3 comes bundled with Mac OS 10.5, testing done with Apple iCal 6.0. (As newer versions become available, they are tested. It is recommended to use these new versions.)
Apple iPhone OS calendar client	6.0.1	Support for scheduling in iOS 4.x on iPhone, iPod touch, and iPad (iOS 8.3 is the latest version).
Lightning	1.9.1	Lightning 4.0.0.1 is the latest version.

Other standard CalDAV clients that should work include eM, Enterprise Tasks, Android CalDAV-Sync, Android CalendarSync, DAVDroid Bitfire, and Notifylink.

Hardware Requirements

The number and configuration of the systems that you employ for your Calendar Server installation depends on the scale and the kind of deployment you have planned.

Note:

The sizing estimates in this section assume proper application configuration and tuning, in a manner consistent with leading practices of Oracle Communications consulting and performance engineering. This information is provided for informational purposes only and is not intended to be, nor shall it be construed as a commitment to deliver Oracle programs or services. This document shall not form the basis for any type of binding representation by Oracle and shall not be construed as containing express or implied warranties of any kind. You understand that information contained in this document will not be a part of any agreement for Oracle programs and services. Business parameters and operating environments vary substantially from customer to customer and as such not all factors, which may impact sizing, have been accounted for in this documentation.

Table 3-5 provides the minimum hardware requirements for Calendar Server deployed on a single managed server in a WebLogic or GlassFish domain.

Table 3-5 Minimum Hardware Requirements

Component	Requirement
Disk Space	Approximately 20 MB required for Calendar Server software.

Table 3-5   (Cont.) Minimum Hardware Requirements

Component	Requirement
RAM	8 GB

Information Requirements

During the Calendar Server installation, you must enter values for configuration items such as host names and port numbers. This section describes the information that you must provide during the installation and initial configuration process.

Calendar Server Information

Table 3-6 lists the Calendar Server information that you provide during initial configuration.

Table 3-6   Calendar Server Information

Information Type	Default Value	Notes
Directory to store configuration and data files	**/var/opt/sun/comms/davserver**	NA
Runtime user ID under which Calendar Server runs	**root**	If you select GlassFish Server to deploy Calendar Server: • The root user must match the user that runs the GlassFish Server instance. This user is also the owner of the application data and configuration directory. If you select WebLogic Server to deploy Calendar Server: • The Runtime user ID must have permission and write access to the Oracle WebLogic Server instance and configuration directory. This ID is also the owner of the application data and configuration directory. Refer to the "Installing and Configuring WebLogic Server" section for setting up WebLogic Server for Calendar Server. **Note**: Only non-root users can install WebLogic Server. Therefore, a non-root system user, for example, **uadmin**, should be created and used for installing, owning, or running the WebLogic Server instances. Ensure that the non-root user system account, which runs the WebLogic Server and the runtime user ID system account belong to the same system user group.

Table 3-6 (Cont.) Calendar Server Information

Information Type	Default Value	Notes
Runtime group to contain Calendar Server runtime user ID	**bin**	If you select GlassFish Server to deploy Calendar Server: • This group must match the group of the user that runs the GlassFish Server instance. This group is also the owner of the application data and configuration directory. If you select WebLogic Server to deploy Calendar Server: • This group must match the group of the user that runs the WebLogic Server instance. This group is also the owner of the application data and configuration directory. For example, if the WebLogic Server instance is owned by a system user/group as **uadmin**/**staff**, the same group **staff** must be used here for the Runtime group. **Note**: Runtime user ID also belongs to the same system group.
Fully qualified host name of this system	FQDN of host	NA

Database Information

Table 3-7 lists the database information that you provide during initial configuration.

Table 3-7 Back-End Database Information

Information Type	Default Value
The type of Calendar Server database. Possible values are either **mysql** or **oracle**.	**mysql**

MySQL Server Database Information

Table 3-8 lists the database information that you provide during initial configuration if you choose MySQL Server as the database.

Table 3-8 MySQL Server Database Information

Information Type	Default Value
MySQL database server host name	FQDN of host
MySQL database server port number	**3306**
Calendar Server database user	**caldav**
Calendar Server database user password	No default value.
Calendar Server database name	**caldav**

Table 3-8 (Cont.) MySQL Server Database Information

Information Type	Default Value
iSchedule database name	**ischedule**

Oracle Database Information

Table 3-9 lists the database information that you provide during initial configuration if you choose Oracle as the database.

Table 3-9 Oracle Database Information

Information Type	Default Value
Oracle database server host name	FQDN of host
Oracle database server port number	**1521**
Oracle service name	**orcl.***domain_name_of_host*
Calendar Server database user	**caldav**
iSchedule database name	**ischedule**
Calendar Server database user password	No default value.

Document Store Information for Calendar Database

Table 3-10 lists the document store information for the calendar database that you provide during initial configuration.

Table 3-10 Document Store Information for Calendar Database

Information Type	Default Value
The document store type. Possible values are **local**, **dbdocstore**, or **remote**. A value of **dbdocstore** is possible only for Oracle Database.	**local**
The name of the Calendar Server back end with which the document store is associated.	**defaultbackend**
The path to the Calendar Server document store.	**/var/opt/sun/comms/davserver/db**
(Remote document store only) The name of the host where the remote Calendar Server document store is located.	No default value.

Table 3-10 (Cont.) Document Store Information for Calendar Database

Information Type	Default Value
(Remote document store only) The port number for the remote Calendar Server document store.	**8008**

Document Store Information for iSchedule Database

Table 3-11 lists the document store information for the iSchedule database that you provide during initial configuration.

Table 3-11 Document Store Information for iSchedule Database

Information Type	Default Value
The document store type. Possible values are **local**, **dbdocstore**, or **remote**. A value of **dbdocstore** is possible only for Oracle Database.	**local**
The name of the iSchedule database.	**ischedulebackend**
The path to the iSchedule database.	**/var/opt/sun/comms/davserver/db/ischedulebackend**
(Remote document store only) The name of the host where the remote document store is located.	No default value.
(Remote document store only) The port number for the remote document store host.	**8008**

GlassFish Server Information

Table 3-12 lists the GlassFish Server information that you provide during initial configuration.

Table 3-12 GlassFish Server Information

Information Type	Default Value
GlassFish Server installation directory	**/opt/glassfish3/glassfish**
GlassFish Server domain directory	**/opt/glassfish3/glassfish/domains/domain1**
GlassFish Server document root directory	**/opt/glassfish3/glassfish/domains/domain1/docroot**
GlassFish Server target instance name	**server**

Table 3-12 (Cont.) GlassFish Server Information

Information Type	Default Value
GlassFish Server virtual server	**server**
GlassFish Server administration server host	FQDN of host
GlassFish Server administration server port	**4848**
Is administration server port secure	**true** (yes)
GlassFish Server administrator user	**admin**
GlassFish Server administrator user password	No default value.
URI path of the deployed server	**https://***FQDN of host*: **443** / (root directory) For information about using a **.well-known** URI, such that access to / (root) or **/.well-known/caldav/** is redirected to the **/dav/principals/** URI, see the topic on Enabling CalDAV and CardDAV Autodiscovery in *Calendar Server System Administrator's Guide*.

WebLogic Server Information

Table 3-13 lists the Oracle WebLogic Server information that you provide during initial configuration.

Table 3-13 WebLogic Server Information

Information Type	Description
WebLogic Server installation directory	Directory in which WebLogic Server is installed. For example, **WLS_HOME/Oracle_Home**
WebLogic Server domain directory	The directory in which domain directories are created. For example: **WLS_HOME/Oracle_Home**/**user_projects/domains/***domain1*
WebLogic Server document root directory	The WebLogic Server document root directory. For example: **WLS_HOME/Oracle_Home/user_projects/domains**/*domain1*.
WebLogic Server target instance name	The name of the WebLogic Server's Managed Server target name. For example: server1
WebLogic Server virtual server	The name of the WebLogic Server's Managed Server target name. For example: server1
WebLogic Server administration server host	FQDN of host
Is administration server port secure	Whether Oracle WebLogic Server administration server port is running over SSL. Default: Enabled
WebLogic Server administration server port	**7001** This is the port with which you log in to WebLogic Administration Server. **Note**: If you set **Is administration server port secure** to **true**, this must be the SSL port of WebLogic Administration Server.

Table 3-13 (Cont.) WebLogic Server Information

Information Type	Description
WebLogic Server administrator user	The admin user name to log in to WebLogic Server Administration Server.
WebLogic Server administrator user password	The password to log in to WebLogic Server Administration Server.
URI path of the deployed server	https://FQDN of host:7003/davserver Pattern should be https://*FQDN of host*:*ManagedServer_Port context-root* ManagedServer_Port: WebLogic Server domain must be set up with at least one ManagedServer which hosts Calendar Server. You must provide the port of that Managed Server here. **Note**: If you select Secure mode, ensure to enter **https** and the SSL port of that Managed Server instance. **context-root**: The default **/davserver** is an example of the context to deploy Calendar Server. For more information, refer to the Enabling CalDAV and CardDAV Autodiscovery section in *Calendar Server System Administrator's Guide*.

LDAP Information

Table 3-14 lists the LDAP information that you provide during initial configuration.

Table 3-14 LDAP Information

Information Type	Default Value
User/Group LDAP URL	**ldaps://***FQDN of host***:636**
User/Group directory manager distinguished name (DN)	**cn=Directory Manager**
Directory manager password	No default value.
LDAP unique ID attribute	**davuniqueid**
User/Group default domain	The name of the domain in the directory user/group tree where user and group objects reside.
Default organization distinguished name (DN)	This value is generated based on the previous value and the user/group suffix of the Directory Server.
Calendar Server administrator user	**calmaster**
Calendar Server administrator user password	No default value.

Notification Information

Table 3-15 lists the email notification information that you provide during initial configuration.

Table 3-15   Notification Information

Information Type	Default Value
Notification mail server host name	FQDN of host
Notification mail server port number	**25**

4

Calendar Server Pre-Installation Tasks

This chapter describes the pre-installation tasks that you must complete before you can install Oracle Communications Calendar Server.

Pre-installation and configuration tasks include:

- Installing Java
- Installing and Configuring GlassFish Server
- Installing and Configuring WebLogic Server
- Installing Directory Server
- Installing the Database

Installing Java

The application server is a Java application and it requires a Java environment in which to run. The Java version must be chosen based on the JDK support available for the container.

The 32-bit and 64-bit JDKs require manual installation. Install both JDKs instead of JRE on your front-end hosts.

Download Java from the Oracle website:

`http://www.oracle.com/technetwork/java/javase/downloads/index.html`

> **Note:**
>
> If you use GlassFish Server 3.x, it requires JDK 1.7. If you use WebLogic Server 12.x, it requires JDK 1.8.

Installing and Configuring GlassFish Server

To install and configure GlassFish Server, see *Oracle GlassFish Server 3.1.2 Installation Guide* at:

`http://docs.oracle.com/cd/E26576_01/doc.312/e24935/installing.htm#ggssq`

Installing and Configuring WebLogic Server

Before you install and configure Oracle WebLogic Server for Calendar Server, prepare the System user and groups for the installation:

- Create a system user and group for Oracle WebLogic Server setup.

> **Note:**
>
> You must install Oracle WebLogic Server by using a non-root user. For example, to install Oracle WebLogic Server, you can use a Unix non-root user as **uadmin** and a Unix group user as **staff**. You can install Calendar Server by using a root user or any other user who is added to the **staff** group and possess the same permissions or access as **uadmin**.

- Create an Oracle WebLogic Server home directory for installation and ensure that the permissions for the required setup directories are set as shown in the following example:
  - mkdir *WebLogic_home*
  - chmod 755 *WebLogic_home*
  - chown -R uadmin *WebLogic_home*
  - chgrp -R staff *WebLogic_home*
- Install JDK 1.8.0 update version on the platform. Ensure **JAVA_HOME** and PATH environment variables are set in the user environment

To install and configure Oracle WebLogic Server for Calendar Server:

1. Download and unzip the ZIP file that you have obtained for the **Generic** package. See `Oracle WebLogic Server Installers` for information on Oracle WebLogic Server download location.
2. Create a Domain, Administration Server, and Managed Server. See `Configuring the WebLogic Domain` for more information.

   See the following Oracle WebLogic Server resources for more information:

   - `Oracle WebLogic Server 12.2.1.3 documentation`
   - `Starting the Installation Program`
   - `A Oracle Universal Installer Installation Screens`
3. Navigate to the WebLogic Domain's bin directory that you have created. For example, *WebLogic_home*/**user_projects**/**base_domain**/**bin**.
4. Modify **setDomainEnv.sh** to add the following applicable settings.
   - The following modification is only for Solaris 11.4:

     ```
     JAVA_OPTIONS="${JAVA_OPTIONS} -Dsun.security.pkcs11.enable-solaris=false"

     export JAVA_OPTIONS
     ```
   - (Optional) If you get an error related to random number when you restart the server, add the following:

     ```
     JAVA_OPTIONS="${JAVA_OPTIONS} -Djava.security.egd=file:/dev/./urandom"

     export JAVA_OPTIONS
     ```
   - To disable DERBY, add the following settings at the end of the file:

     ```
     DERBY_FLAG="false"

     export DERBY_FLAG
     ```
5. Ensure to set the environment in the terminal that is used to start the servers by sourcing the **setDomainEnv.sh** file as shown below:

```
cd WebLogic_home/user_projects/base_domain/bin

source ./setDomainEnv.sh or . ./setDomainEnv.sh
```

6. Start the Administration Server.
7. Configure the Managed Server for default HTTP and HTTPS ports and start the Managed Server.
8. Configure Oracle WebLogic Server in a secure mode using the following details:

   To enable SSL and configure keystores in Oracle WebLogic Server:

   - See the Oracle WebLogic Server documentation at:

     https://docs.oracle.com/middleware/12213/wls/SECMG/identity_trust.htm#SECMG365

   - Oracle WebLogic Server offers four keystore options in its configuration. However, only the following keystore options are recommended for Calendar Server:
     - CustomIdentityandCustomTrust
     - CustomIdentityandJavaStandardTrust

     > **Note:**
     >
     > You must always set the keystore type to **JKS**.

   - The keystore configuration must be same for an Administration Server and Managed Servers. It means, you should configure the same options on both servers for hosting Calendar Server.
   - You must set keystore passwords identical to the Oracle WebLogic Server Administration Sever password.

     > **Note:**
     >
     > Calendar Server is deployed on Oracle WebLogic Server securely only if the keystore passwords and Oracle WebLogic Server password match.

9. Ensure that the Administration Server and Managed Server are started successfully.

   You must perform the following post configuration steps before the product configuration. These steps bypass the default basic auth model of WebLogic Server and ensure the product's own basic auth security model. Otherwise, WebLogic Server causes a double login prompt.

   a. Shutdown WebLogic Server (both Admin Server and Managed Server).

   > **Note:**
   >
   > WebLogic Server must not be running while modifying **config.xml**. If WebLogic Server runs, changes will be overwritten.

   b. Open a terminal and go to the WebLogic Server domain's config directory:

```
Weblogic_Domain/config
```

c. Open **config.xml**, search for the **security-configuration** section, and then add the following XML code into the **security-configuration** node as shown below:

```
....
<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-
credentials>
</security-configuration>
```

d. Ensure that permissions are set correctly again, similar to how they were set during the installation (to ensure permissions are not modified while editing files). If not, set using the following commands:

```
chown -R uadmin WebLogic_home
chgrp -R staff WebLogic_home
```

e. Restart the WebLogic Server. Ensure AdminServer and ManagedServer are successfully running.

Installing Directory Server

Calendar Server uses Directory Server to store and access LDAP data for individual users, groups, and domains.

If your site does not currently have Directory Server deployed, and you need to install either Oracle Directory Server Enterprise Edition or Oracle Unified Directory (depending on the Calendar version you are using). For more information, see:

- ODSEE: ODSEE Documentation
- OUD: OUD Documentation

Prior to installing and configuring Calendar Server, you must also prepare the Directory Server LDAP schema by running the either **comm_dssetup.pl** or **rundssetup** script (depending on the Calendar version you are using). This script, which is provided as part of the Calendar Server media pack, adds the necessary Communications Suite schema to the LDAP. See "Preparing Directory Server" for more information.

Some LDAP object classes in the Communications Suite schema specifically support Calendar Server. To understand the schema that is used by Calendar Server, refer to *Communications Suite Schema Reference*.

The **davcore.ldapattr.*** configuration parameters govern the default values for LDAP attributes and object classes used by Calendar Server. Default values are set based on the Communications Suite schema. See the topic on Calendar Server configuration parameters in *Calendar Server System Administrator's Guide* for more information.

Installing the Database

You can use either Oracle Database or MySQL Server as the Calendar Server database (the calendar store). You cannot mix database types in a deployment.

If you have a multiple host deployment, you can install all databases and complete all database preparation before installing Calendar Server. When you configure Calendar Server by running the **init-config** script, you choose one to be the primary database. You then configure the other databases as described in "Configuring Calendar Server with Multiple Hosts".

Depending on your database choice, go to one of the following tasks:

- Installing and Configuring MySQL Server
- Installing and Creating an Oracle Database Instance for Calendar Server

Installing and Configuring MySQL Server

To install and configure MySQL Server:

1. Download the MySQL Server software from My Oracle Support, at:

   ```
   http://support.oracle.com
   ```

   See "Required Software" for information about supported versions of MySQL Server.

2. As **root**, add the new MySQL package.

   > **Note:**
   >
   > Adding the new package on Red Hat Linux might cause the system to automatically run the **mysql_upgrade** command. This is fine in the context of this procedure.

   - On Solaris:

     ```
     pkgadd -d mysql-advanced-5.x.xx-solaris11-sparc-64bit.pkg
     ```

   - On Oracle Linux :

     ```
     rpm --ivv MySQL-server-advanced-5.x.xx-1.el6.x86_64.rpm
     rpm --ivv MySQL-client-advanced-5.x.xx-1.el6.x86_64.rpm
     ```

   Linux has two MySQL RPMs, client and server, that you must add.

3. If the MySQL **/etc/my.cnf** configuration file is absent, create it with the following content:

   - Solaris OS:

     ```
     [mysqld]
     basedir = /opt/mysql/mysql
     datadir = /var/mysql
     default-storage-engine = InnoDB
     character-set-server = utf8mb4
     transaction-isolation = READ-COMMITTED
     ```

   - Red Hat Linux or Oracle Linux:

     ```
     [mysqld]
     basedir = /usr
     datadir = /var/mysql
     default-storage-engine = InnoDB
     character-set-server = utf8mb4
     transaction-isolation = READ-COMMITTED
     ```

   Change the value of **basedir** and **datadir** if needed. Make sure there are no extra spaces if you cut and paste the path.

4. If you use replication, use ROW binary logging, instead of the default STATEMENT logging, by adding the following line to the **/etc/my.cnf** file:

   ```
   binlog-format=ROW
   ```

5. Set values for cache size, max connection size, and other parameters that impact MySQL Server performance.

For example, see the MySQL Server Tuning information in *Calendar Server System Administrator's Guide*.

6. Remove the **log_long_format** entry from the **/etc/my.cnf** file. Use of the **log_long_format** is no longer a valid configuration option in 5.5. If you do not remove it, MySQL Server does not start.
7. Start MySQL Server:

   ```
   /etc/init.d/mysql start
   ```

Continue with the "Installing Calendar Server" section.

Installing and Creating an Oracle Database Instance for Calendar Server

You can install an Oracle Database for Calendar Server in one of the following ways:

- Installing a New Oracle Database
- Creating a New Database From an Existing Oracle Installation
- Using an Existing Database for the Calendar Server Schema

> **Note:**
>
> The Oracle Database instance must be a Unicode database, that is, the database character set must be AL32UTF8. Calendar Server does not support non-Unicode database character sets.

Installing a New Oracle Database

You install and create the Oracle Database instance by using the Oracle Universal Installer.

1. Download Oracle Database from the Oracle Technology Network website at:

   ```
   https://www.oracle.com/database/technologies/oracle-database-software-downloads.html
   ```

2. To install and create a new Oracle Database instance, follow the instructions in the installation guide for Oracle Database for your operating system, at:
   - Oracle Database 11g, Release 2:

     https://docs.oracle.com/cd/E11882_01/install.112/e47689/title.htm
   - Oracle Database 12:

     `http://docs.oracle.com/database/121/nav/portal_11.htm`
3. When installing and creating the database instance, follow these guidelines:
   - On UNIX and Linux installations, ensure that you set the directory owner, group, and permissions correctly.
   - When selecting the database character set in the Specify Database Configuration Options window, select **UTF-8 AL32UTF8**. Deselect the **Create database with sample schema** option in the Database Example section.
   - When choosing the database management method in the Select Database Management Options window, select **Use Database Control for Database Management**.

Continue with "Installing Calendar Server".

Creating a New Database From an Existing Oracle Installation

You create a new Oracle Database instance by using the Oracle Database Configuration Assistant (DBCA).

1. To create an Oracle Database instance, follow the instructions in the topic on creating and managing a database with DBCA in *Oracle Database 2 Day DBA* documentation, at:
   - Oracle Database 12c:

     `https://docs.oracle.com/database/121/ADMIN/create.htm#ADMIN002`
   - Oracle Database 11g, Release 2:

     `http://docs.oracle.com/cd/E11882_01/server.112/e10897/install.htm#ADMQS0232`
2. When creating the database instance, follow these guidelines:
   - When choosing the template, select the default **General Purpose**.
   - When choosing the database management method in the Select Database Management Options window, select **Use Database Control for Database Management**.
   - When choosing the storage options, select **File System**.
   - When choosing database content, deselect the sample schema.
   - When selecting the database character set in the Specify Database Configuration Options window, select **UTF-8 AL32UTF8**.

Continue with "Installing Calendar Server".

Using an Existing Database for the Calendar Server Schema

You can use an existing Oracle database for the Calendar Server schema if the database is configured to use the AL32UTF8 Unicode character set. If the database uses the Unicode character set, you prepare the database for use with Calendar Server after installing Calendar Server.

You can check whether the Oracle Database instance is running with the AL32UTF8 database character set by typing the following query using Oracle SQL*Plus SQL Plus:

```
SELECT * FROM NLS_DATABASE_PARAMETERS WHERE PARAMETER LIKE 'NLS_CHARACTERSET';
```

Examine the results.

If the query returns AL32UTF8, the database can be used for creating Calendar Server schema. Continue with "Installing Calendar Server".

If the query does not return AL32UTF8, you must either create a new database that uses AL32UTF8 or migrate the database character set to AL32UTF8. Database character set migration is usually an intricate process that requires careful planning and implementation strategies. For details, consult *Oracle Database Globalization Support Guide* for the character set migration chapter, and documentation for *Oracle Database Migration Assistant for Unicode*. To create a new database instance, follow the instructions in the "Using an Existing Database for the Calendar Server Schema" section.

5

Installing Calendar Server

This chapter describes how to install and configure Oracle Communications Calendar Server.

Before installing Calendar Server, read these chapters:

- Calendar Server Installation Overview
- Planning Your Calendar Server Installation
- Calendar Server System Requirements
- Calendar Server Pre-Installation Tasks

Installation Assumptions

The instructions in this chapter assume:

- You are deploying Calendar Server on a single host or Solaris zone, or multiple hosts or Solaris zones.
- Directory Server is already installed.
- You have installed and configured the application server as the web container for Calendar Server.
- You have installed the database back end for storing the calendar information.

Installing Calendar Server

The tasks to install Calendar Server are as follows:

- Downloading the Calendar Server Software
- Preparing Directory Server
- Installing the Calendar Server Software

Downloading the Calendar Server Software

1. Download the Calendar Server software, and the Oracle Communications Directory Server Setup for your operating system from the Oracle software delivery website, located at:

   `http://edelivery.oracle.com/`

2. Copy the Calendar Server ZIP file to a temporary directory on your Calendar Server hosts and extract the files, to be enable to install the Calendar Server software.
3. Copy the Directory Server Setup ZIP file to a temporary directory on your Directory Server hosts and extract the files, to be able to install and run the either **comm_dssetup.pl** or **rundssetup** script (depending on the Calendar version you are using).

Preparing Directory Server

You can prepare your Directory Server by running either the **comm_dssetup.pl** or **rundssetup** script (depending on the Calendar version you are using) against it. You can run either of the scripts in interactive or silent mode. For instructions to run the **rundssetup** script in silent mode, see "Running the comm_dssetup.pl Script in Silent Mode" or "Running the rundssetup Script in Silent Mode".

Running the comm_dssetup.pl Script in Interactive Mode

To prepare Directory Server and run the **comm_dssetup.pl** script in interactive mode:

1. On the host where Directory Server is installed, log in as or become the superuser **(root)**.
2. Start Directory Server, if necessary.
3. Change to the directory where you extracted the Directory Server Setup ZIP file and run the installer.

   ```
   commpkg install
   ```

   For more information about running the installer, see "commpkg Reference".
4. Select **Comms DSsetup** and proceed with the installation.
5. Run the **comm_dssetup.pl** script in interactive mode (without any arguments), then enter your choices when prompted.

   ```
   /usr/bin/perl comm_dssetup.pl
   ```

   For more information, see "comm_dssetup.pl Reference".

   > **Note:**
   >
   > You can use either LDAP Schema 2 or Schema 1.

6. If necessary, provision users in the Directory Server.

   If Directory Server is already installed at your site, users have already been provisioned. If you have just installed Directory Server at your site, then you need to provision users. For information, see the discussion on provisioning users and schema in the *Schema Reference*.

Installing the Calendar Server Software

To install the Calendar Server software:

1. On the Calendar Server host, log in as or become the superuser **(root)**.
2. Go to the directory where you extracted the Calendar Server files.
3. Run the installer.

   ```
   commpkg install
   ```

   For more information about running the installer, see "commpkg Reference".
4. Select **Calendar Server** and proceed with the installation.

5. (Optional) If you are going to use remote document stores, install the Calendar Server software on the remote hosts.

When the installation is complete, depending on your database choice, go to one of the following tasks:

- Preparing MySQL Server for Use with Calendar Server
- Preparing Oracle Database for Use with Calendar Server

Preparing MySQL Server for Use with Calendar Server

You can either perform the following steps in this section or run the **config-mysql** script to prepare a new MySQL installation for use with Calendar Server. See "Running the config-mysql Script" for more information.

> **Note:**
>
> When necessary, the following instructions show where commands differ between Solaris OS, and Red Hat Linux or Oracle Linux.

> **Caution:**
>
> You must use the MySQL Connector version packaged with Calendar Server. Do not use a different version.

To prepare MySQL Server for use with Calendar Server:

1. Create a **mysql** group. For example:

   ```
   /usr/sbin/groupadd mysql
   ```

2. Create a **mysql** user by running the following command.

   ```
   /usr/sbin/useradd -g mysql mysql
   ```

3. Initialize the database.

   a. Remove the pre-created data. For example:

      ```
      rm -rf /opt/mysql/mysql/datarm -rf /var/lib/mysql
      ```

   b. Create an initial database using the following command, substituting a different directory for **/var/mysql** if desired.

      Solaris OS:

      ```
      /opt/mysql/mysql/scripts/mysql_install_db --user=mysql --ldata=/var/mysql
      ```

      Red Hat Linux or Oracle Linux:

      ```
      /usr/bin/mysql_install_db --user=mysql --ldata=/var/lib/mysql
      ```

4. Verify that the MySQL configuration file, **/etc/my.cnf**, exists with the following content:

   - Solaris OS:

```
[mysqld]
basedir = /opt/mysql/mysql
datadir = /var/mysql
default-storage-engine = InnoDB
character-set-server = utf8mb4
transaction-isolation = READ-COMMITTED
```

- Red Hat Linux or Oracle Linux:

  ```
  [mysqld]
  basedir = /usr
  datadir = /var/lib/mysql
  default-storage-engine = InnoDB
  character-set-server = utf8mb4
  transaction-isolation = READ-COMMITTED
  ```

Change the value of **basedir** and **datadir** if needed. Make sure there are no extra spaces if you cut and paste the path.

5. Install the startup script:

   - Solaris OS:

     ```
     cp /opt/mysql/mysql/support-files/mysql.server /etc/init.d/mysql
     ```

   - Red Hat Linux or Oracle Linux:

     ```
     cp /usr/share/mysql/mysql.server /etc/init.d/mysql
     ```

6. Start MySQL Server.

   ```
   /etc/init.d/mysql start
   ```

7. Change the MySQL **root** password.

   - Solaris OS:

     ```
     /opt/mysql/mysql/bin/mysqladmin -u root password 'password'
     ```

   - Red Hat Linux or Oracle Linux:

     ```
     /usr/bin/mysqladmin -u root password 'password'
     ```

     Replace *password* with the appropriate password to use for the **root** user.

8. Run the secure MySQL installation to disable remote root access, remove anonymous users, remove test databases, and so on.

   - Solaris OS:

     ```
     /opt/mysql/mysql/bin/mysql_secure_installation
     ```

   - Red Hat Linux or Oracle Linux:

     ```
     /usr/bin/mysql_secure_installation
     ```

9. Create the MySQL user, and caldav and iSchedule databases.

   The following example uses **caldav** as the MySQL user name, **caldav** as the database name, and **ischedule** as the iSchedule database name. You can use different names. For more information, see the MySQL CREATE USER syntax at:

   http://dev.mysql.com/doc/refman/5.5/en/create-user.html

   ```
   /opt/mysql/mysql/bin/mysql -u root -p
   Enter password:
   mysql> CREATE DATABASE caldav CHARACTER SET = utf8mb4;
   mysql> CREATE USER 'caldav'@'localhost' IDENTIFIED BY 'password';
   mysql> GRANT ALL ON caldav.* TO 'caldav'@'localhost';
   mysql> CREATE DATABASE ischedule CHARACTER SET = utf8mb4;
   ```

```
mysql> GRANT ALL ON ischedule.* TO 'caldav'@'localhost';
mysql> exit
```

Replace *localhost* with the Calendar Server host name if MySQL Server is on a remote host (or use **%** for any host). Also, replace *password* with the password to use for the **caldav** user.

10. Do one of the following to configure MySQL to start automatically upon system restart.
    - Create a symbolic link to the MySQL start script.

      ```
      ln /etc/init.d/mysql /etc/rc3.d/S99mysql
      ```

      If must add the stop scripts, use the following command:

      ```
      ln /etc/init.d/mysql /etc/rc0.d/K48mysql
      ```

    - Configure MySQL to run with Solaris Management Facility (SMF). See the following links for information about how to configure MySQL to run with Solaris Management Facility (SMF):

      https://blogs.oracle.com/ritu/entry/how_to_configure_mysql_to

      http://www.oracle.com/technetwork/systems/articles/smf-example-jsp-136458.html

      https://blogs.oracle.com/smenon/entry/how_to_configure_mysql_to

After preparing MySQL Server for use with Calendar Server, continue with the "Configuring Calendar Server" section.

Preparing Oracle Database for Use with Calendar Server

Use the appropriate task based on your Oracle Database version:

- Preparing Oracle Database 11g Release 2 or Oracle Database 12c Not Pluggable (non-CDB)
- Preparing Oracle Database 12c Container Database

Preparing Oracle Database 11g Release 2 or Oracle Database 12c Not Pluggable (non-CDB)

To prepare Oracle Database 11g Release 2 or Oracle Database 12c not pluggable (non-CDB):

1. If your Oracle Database resides on a different host from where you installed the Calendar Server software, copy the **config-oracle** script and the **Util.pm** module, located in the *CalendarServer_home***/tools/unsupported/bin** directory, to the Oracle Database host.
2. Ensure that the Oracle environment has been configured after installing the software by running the **/usr/local/bin/oraenv** script (or **corenv** for C Shell).
3. Run the **config-oracle** script.

   ```
   cd CalendarServer_home/tools/unsupported/bin
   config-oracle -c -i
   ```

4. Respond to the following prompts.

   ```
   Enter the Oracle SYS user password:
   Enter the name of the Calendar db user (caldav):
   Enter the password for Calendar db user user:
   Please enter password again:
   Enter the name of the iSchedule db user (ischedule):
   ```

```
Enter the password for iSchedule db user user_ischedule:
Please enter password again:
```

The script configures the Oracle Database for use with Calendar Server. See "Running the config-oracle Script" for more information.

After preparing Oracle Database for use with Calendar Server, continue with the "Configuring Calendar Server" section.

Preparing Oracle Database 12c Container Database

You cannot run the **config-oracle** script to prepare an Oracle Database 12c container database (that is, one that uses a pluggable database) Instead, you must manually prepare Oracle Database for Calendar Server:

1. Ensure that the Oracle environment has been configured after installing the software by running the **/usr/local/bin/oraenv** script (or **corenv** for C Shell).
2. Run the following commands to prepare Oracle Database for Calendar Server:

```
sqlplus / as sysdba
       ALTER SESSION SET CONTAINER = PDBORCL;
       CREATE USER caldav IDENTIFIED BY password;
       GRANT CONNECT, RESOURCE TO caldav;
       GRANT CREATE VIEW to caldav;
       GRANT UNLIMITED TABLESPACE to caldav;
       GRANT EXECUTE ON DBMS_LOCK to caldav;
```

3. Run the following commands to prepare the iSchedule database:

```
sqlplus / as sysdba
       ALTER SESSION SET CONTAINER = PDBORCL;
       CREATE USER ischedule IDENTIFIED BY password;
       GRANT CONNECT, RESOURCE TO ischedule;
       GRANT CREATE VIEW to ischedule;
       GRANT UNLIMITED TABLESPACE to ischedule;
       GRANT EXECUTE ON DBMS_LOCK to ischedule;
```

After preparing Oracle Database for use with Calendar Server, continue with the "Configuring Calendar Server" section.

Installing Calendar Server in Silent Mode

When you run the Calendar Server installer in silent mode, you are running a non-interactive session. The installation inputs are taken from the following sources:

- A silent installation file (also known as a state file)
- Command-line arguments
- Default settings

You can use silent mode to install multiple instances of the same software component and configuration without having to manually run an interactive installation for each instance.

This section includes:

- Performing a Calendar Server Silent Installation
- About Upgrading Shared Components
- Silent Mode File Format

Performing a Calendar Server Silent Installation

To perform a Calendar Server silent installation:

1. Obtain the state file by one of the following means.
   - Run an interactive installation session and use the state file that is created in the **/var/opt/CommsInstaller/logs/** directory. The state file name is similar to **silent_CommsInstaller_20070501135358**. A state file is automatically created for every run of the installation.
   - Create a silent state file without actually installing the software during the interactive session by using the **--dry-run** option, then modifying the state file. For example:

     ```
     commpkg install --dry-run
     ```
2. Copy the state file to each host machine and edit the file as needed. See "Silent Mode File Format".
3. Run the silent installation on each host. For example:

   ```
   commpkg install --silent input_file
   ```

   where *input_file* is the path and name of the silent state file, for example **/var/opt/CommsInstaller/logs/silent_CommsInstaller_20070501135358**.

   For details about the **--silent** option, see "install Verb Syntax".

   > **Note:**
   >
   > Command-line arguments override the values and arguments in the state file.

About Upgrading Shared Components

By default, shared components that require user acceptance for upgrading are not upgraded when you run a silent installation. The option to upgrade shared components in the silent state file is automatically disabled. That is, the option is set to **UPGRADESC=No**. This is true even if you explicitly asked to upgrade shared components when you ran the interactive installation that generated the silent state file. That is, you ran either **commpkg install --upgradeSC y** or you answered “yes" when prompted for each shared component that needed upgrading.

Disabling upgrading shared components in the silent state file is done because the other hosts on which you are propagating the installation might have different shared components installed, or different versions of the shared components. Therefore, it is safer to not upgrade the shared components by default.

You can upgrade shared components when you run a silent installation by performing either of the following actions:

- Use the **--upgradeSC y** option when you run the silent installation. (The command-line argument overrides the argument in the state file.)
- Edit the **UPGRADESC=No** option in the silent state file to: **UPGRADESC=Yes**.

> **Caution:**
>
> If you do not upgrade shared components your installation might not work properly.

Silent Mode File Format

The silent mode file (also known as a state file) is formatted like a property file: blank lines are ignored, comment lines begin with a number sign (#), and properties are key/value pairs separated by an equals (=) sign. Table 5-1 shows which options you can change and provides examples:

Table 5-1 Silent Mode File Options

Option	Description	Example
VERB	Indicates which function to perform. For a silent install, this is set to **install**.	VERB=install
ALTDISTROPATH	Indicates an alternate distro path.	ALTDISTROPATH=SunOS5.10_i86pc_DBG.OBJ/release
PKGOVERWRITE	A boolean indicating whether to overwrite the existing installation packages. (See the **--pkgOverwrite** switch).	PKGOVERWRITE=YES
INSTALLROOT	Specifies installation root.	INSTALLROOT=/opt/sun/comms
ALTROOT	A boolean indicating whether this is an alternate root install.	ALTROOT=yes
EXCLUDEOS	Specifies to not upgrade operating system patches.	EXCLUDEOS=YES
EXCLUDESC	Specifies to exclude shared component patches.	EXCLUDESC=no
COMPONENTS	A space separated list of mnemonics of the components to be installed. You can precede the mnemonic with a ~ to indicate that only the shared components for that product be installed.	COMPONENTS=CS
ACCEPTLICENSE	This option is no longer used.	Not applicable
UPGRADESC	Indicates whether all shared components should or should not be upgraded without prompting.	UPGRADESC=no
INSTALLNAME	The friendly name for the INSTALLROOT.	INSTALLNAME=
COMPONENT_VERSIONS	This option is unused.	Not applicable

To display a complete list of the product names (such as MS, MS64, CS) to use with the **COMPONENTS** property, run the **commpkg info --listPackages** command. This command displays the mnemonics for each product. For more information, see the discussion on the **commpkg info** command in "commpkg Reference".

Installing Calendar Server on Solaris Zones

This information explains how to install Calendar Server on Solaris OS Zones.

The topics in this section include:

- Installing on Solaris OS Zones: Best Practices
- Installing into a Non-Global Whole Root Zone
- Installing into a Non-Global Sparse Root Zone

Installing on Solaris OS Zones: Best Practices

You can install Calendar Server in the global zone, whole root non-global zones, and sparse non-global zones. Follow these guidelines:

- Treat the global zone as an “administration zone."

  Install shared components and OS patches in the global zone that are to be shared among all zones. However, do not install and run products from the global zone.
- Use whole root non-global zones to run Calendar Server.

  Do not use the global zone or sparse zones. A whole root zone can have versions that are different from other whole root zones, thus giving it a measure of being “self-contained."

Be aware of the following zone aspects:

- You can have different shared component versions in the whole root non-global zone, but it isn't entirely insulated. If you do a packaging or patching operation in the global zone for a shared component, that operation is also attempted in the whole root zone. Thus, to truly have different shared component versions, use an alternate root.
- To avoid affecting whole root zones you can attempt to never install and patch shared components in the global zone. However, it might not be realistic to never have to install or patch a shared component in the global zone. For example, NSS is a shared component, but it is part of Solaris OS. So to expect to never install and patch NSS in the global zone seems unrealistic, especially given it is a security component.
- Although it isn't a recommended best practice, you can use Calendar Server in sparse non-global zones. Do note that shared components cannot be installed into the default root because many of them install into the read-only shared file system (**/usr**). Thus, you must run the installer in the global zone to install shared components into the default root. Prepend your selection with **~** in the global zone to install only the dependencies (that is, shared components). You do not have to install in the global zone first before installing in the sparse zone. The installer allows you to continue even when you do not install all the dependencies. However, upgrading the shared components in the global zone affects the sparse non-global zones, thus requiring downtime for all affected zones simultaneously.

> **Note:**
>
> Sparse root zones are not available beginning with Oracle Solaris 11.

Installing into a Non-Global Whole Root Zone

The non-global whole root zone scenario is the equivalent of installing Calendar Server on a single box with no zones. Simply install Calendar Server as you normally would.

> **Caution:**
>
> Any operations performed in the global zone (such as installations, uninstallations, and patching) affect the whole root zones.

Installing into a Non-Global Sparse Root Zone

Although it isn't a recommended best practice, you can use Calendar Server in a non-global sparse root zone on Solaris 10. To install Calendar Server in a non-global sparse root zone, you first need to install or upgrade the applicable OS patches and shared components in the global zone. You are unable to do so in the sparse root zone, because the **/usr** directory (where the shared components reside) is a read-only directory in the sparse root zone.

1. Follow the pre-installation requirements as described in "Calendar Server Pre-Installation Tasks".
2. Verify that you are about to install the shared components and OS patches in the global zone and not the sparse root zone. To verify you are in the global zone, run **zonename**. The output should be global.
3. Run the installer in the global zone and only install or upgrade the OS patches and the Shared Components. Do not install Calendar Server in the global zone. To do this, add a **~** (tilde) to the component number you want to install in the sparse zone.

   For example, if you plan to install Calendar Server in the sparse zone, you select **~1** during the global zone installation. The installer will know to only install dependencies and not the product itself.
4. Once you have the shared components and OS patches installed, install Calendar Server in the sparse root zone.

Configuring Calendar Server

You must configure Calendar Server to complete the installation. You use the Calendar Server configuration command-line script, **init-config**, to perform this initial runtime configuration.

Calendar Server Initial Configuration Prerequisites

Before running the Calendar Server **init-config** script, ensure that you have satisfied the following prerequisites:

- On the Directory Server host, you ran the **rundssetup** script, as described in "Preparing Directory Server".
- (Linux only) On Calendar Server front-end hosts, install the **openldap-clients** RPM, which installs LDAP tools such as **/usr/bin/ldapsearch** and **/usr/bin/ldapmodify**.

Validating and Storing Oracle WebLogic Server SSL Details

> **Note:**
>
> If you are configuring Calendar Server with WebLogic Server in a secure-mode, ensure to perform the steps provided in the "Installing and Configuring WebLogic Server" section and keep the keystore details handy.

When you configure Calendar Server for the first time with Oracle WebLogic Server in a secure mode, run the **extractSSLArgs.sh** script. This script validates the SSL configuration details in Oracle WebLogic Server and stores the valid details in a format that is required by Calendar Server for all future deployments and processing.

To validate and store Oracle WebLogic Server SSL details for Calendar Server in a secure mode:

1. Open a new terminal and prepare the terminal by sourcing the **setDomainEnv.sh** script of the Oracle WebLogic Server domain:

   ```
   cd Weblogic_Domain/bin

   source ./setDomainEnv.sh   OR  . ./setDomainEnv.sh
   ```

2. Set the **WLST_PROPERTIES** environment variable depending on the selected Oracle WebLogic Server keystore configuration.
   - If the **CustomIdentityandCustomTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_PROPERTIES="-Dweblogic.security.TrustKeyStore=CustomTrust , -
     Dweblogic.security.CustomTrustKeyStoreFileName= WebLogic_home/user_projects/
     domains/base_domain/mytrust.jks"
     ```

     where *WebLogic_home/user_projects/domains/base_domain/mytrust.jks* is the location of truststore file.
   - If the **CustomIdentityandJavaStandardTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_PROPERTIES="-Dweblogic.security.TrustKeyStore=JavaStandardTrust"
     ```

3. Run the **extractSSLArgs.sh** bash shell script **extractSSLArgs.sh** which is available under the Calendar Server installed location: *CalendarServer_Installedlocation*/**sbin**:

   ```
   sh ./extractSSLArgs.sh -u weblogic_admin_user -p weblogic_admin_user_password -l
   t3s://weblogic_server_host:SSL_port
   ```

   If the script is successfully run, it creates **.wls_sslargs** in the configuration directory of your Oracle WebLogic Server domain. You can verify the creation of **.wls_sslargs** by navigating to *Weblogic_Domain*/**config**.

Configuring Java for Calendar Server

If you chose a non-root user as the GlassFish Server runtime user, do the following on the Calendar Server front-end hosts so that Calendar Server locates the proper JDK:

1. Create a symbolic link between **/usr/jdk/latest** and the desired installed JDK in the **/usr/jdk** directory. For example:

   ```
   ln -s /usr/jdk/jdk1.7.0_25 /usr/jdk/latest
   ```

2. Define the **JAVA_HOME** variable in the GlassFish Server user's login profile. You cannot only define the variable in the current shell that you are using to run the **init-config** script.

   If the GlassFish Server user is referencing a JDK in a different location, set that location in the user's **.profile** file by adding the following line. If you want, you can add the line to the system-wide profile file, **/etc/profile**, instead.

   ```
   export JAVA_HOME=JDK_location
   ```

If you use WebLogic Server:

Ensure the WebLogic Administration Server and Managed Server are correctly set up with JDK1.8.0 update version support. Whenever you start or stop WebLogic Server from the system terminal, ensure the **JAVA_HOME** and PATH environmental variables on that terminal are pointing to the same JDK1.8.0_xx version which was used during the WebLogic Server setup. That is, the symbolic link /usr/jdk/latest, JAVA_HOME must be pointing to the same JDK location.

Running the Calendar Server Initial Configuration Script

To run the Calendar Server initial configuration:

1. Log in as or become the superuser (**root**).

   > **Note:**
   >
   > Log in as "`su -`" when running **init-config** if you installed GlassFish Server in a secure mode.

2. Change to the *CalendarServer_home*/**sbin** directory.

   The default installation directory is, **/opt/sun/comms/davserver**.

3. Run the Calendar Server initial configuration script and respond to the prompts.

   See "Calendar Server Configuration Scripts" for more information.

   > **Note:**
   >
   > See "Information Requirements" for information about the values you must provide during the initial configuration.

4. Run the initial configuration script in an interactive mode.
   - If you use GlassFish Server, run **init-config**
   - If you use WebLogic Server, run **init-config -W**

> **Note:**
>
> If you do not use the **-W** option, by default, GlassFish Server is used to configure the **init-config** script.
>
> When you configure Calendar Server for the first time with Oracle WebLogic Server in a secure mode, run the **extractSSLArgs.sh** script. This script validates the SSLconfiguration details in Oracle WebLogic Server and stores the valid details in a format that is required by Calendar Server for all future deployments and processing.

5. When prompted, enter the Calendar Server settings:
   - Directory to store configuration and data files
   - Calendar Server runtime user
   - Calendar Server runtime group
   - Fully qualified host name of this system.
6. Enter the Back-End Database Settings.

   Choose MySQL database or Oracle database (the default is **mysql**), then enter the following information, depending on whether you chose MySQL database or Oracle database:
   - MySQL Database Details
     - Server Host
     - Server Port
     - Calendar Database User
     - Calendar Database User Password (The password you provided while creating the **dav** user in "Preparing MySQL Server for Use with Calendar Server".)
     - Calendar Server DB Name
     - iSchedule DB Name
   - Oracle Database Server Details
     - Server Host
     - Server Port
     - Service Name
     - Calendar DB User
     - Password (The password you provided while creating the **caldav** user in "Preparing Oracle Database for Use with Calendar Server".)
     - iSchedule DB User
     - Password

     If you receive a database connection error, you are prompted to re-enter all the database information in this step. When the validation passes, continue with the next step.
7. Enter the Calendar Back-End Document Store Settings.
   - Back-end document store type, either **local**, **dbdocstore**, or **remote** (See "Configuring the Calendar Server Document Store" for more information.)

- Back-end name with which the document store is associated
- Back-end store path

8. Enter the iSchedule Back-End Document Store Settings.
   - Back-end document store type, either **local**, **dbdocstore**, or **remote**.
   - iSchedule back-end name
   - Back-end store path
9. Enter the Application Server Configuration Details.
   - Install Directory
   - Domain Directory
   - Document Root Directory
   - Server Target Name
   - Virtual Server Identifier
   - Application Server administration server host
   - Application Server administration server port
   - Secure Administration Server Instance
   - Administrator User ID
   - Administrator Password
   - Full URI of the deployed server (default: **https://***FQDN of host:443/*)

     Use **/** (root) as the default context URI for production deployments. Using root enables users to configure their clients by typing a host name. If root is not used, users must enter a long URL, which leads to misconfigurations. The following example URIs describe the behavior of this entry:

     – **https:/host.example.com:443/** - Deploy in root.

     – **https:/host.example.com:443/davserver** - Deploy in **davserver**.

     – **https:/host.example.com:443** - Not allowed, deployment context cannot be blank.

     – **https:/host.example.com:443/cs/dav** - Not allowed, deployment context cannot be more than one level.
   - If you use GlassFish Server:

     Using root as the URI path replaces the GlassFish Server's main index.html file. Therefore, the GlassFish Server welcome page does not display at http://host:port. If you deploy Calendar Server by using a root (**/**) URI context, you cannot deploy Convergence in the same instance of GlassFish Server without moving the **/iwc_static/** files to a different location. The Convergence static files are deployed in the default root context and if another application is deployed to **/**, GlassFish Server cannot find the **/iwc_static/*** URLs. This results in 404 error immediately after login.

     Therefore, create an additional domain in GlassFish Server that listens on a different port so that Convergence and Calendar Server run in separate instances. See *Oracle GlassFish Server 3.0.1 Reference Manual* for information on creating a new GlassFish domain. Additionally, for assistance in configuring clients for non-root URL configurations, see the topic on accounts that use non-standard or demo settings in *Calendar Server System Administrator's Guide*.
   - If you use WebLogic Server:

The URI pattern for deploying the server must be:

https://*FQDN of host*:*ManagedServer_Port*/*context-root*

where:

ManagedServer_Port: WebLogic Server domain must be set up with at least one ManagedServer which hosts Calendar Server. The port of that Managed Server must be provided here.

> **Note:**
>
> If you select a secure mode, ensure to enter **https** and the SSL port of the Managed Server instance.

context-root can be **/** or **/davserver**.

10. Enter the User/Group Directory Server Details.

    - LDAP URL

      The default uses LDAP over SSL (`ldaps://`). If you want Calendar Server to communicate by using SSL with Directory Server in the runtime configuration, you must use LDAPS in the URL specification. You must also put the certificate database containing the Directory Server certificate in the *CalendarServer_home***/lib** directory. This setup works only for Solaris OS. See the openldap documentation for the required SSL setup on Linux. If you use LDAP in the URL specification, you need not perform any SSL setup for the initial configuration. However, you can enable LDAP SSL communication in the runtime configuration as described in *Calendar Server Security Guide*.

    - Bind Distinguished Name (DN)
    - Bind Password

      The following message appears if you do not have the correct password:

      ```
      ldap_bind: Invalid credentials (49)
      Error validating password for cn=Directory Manager
      ```

      In this case, you are prompted again to enter the password.

    - LDAP unique ID attribute

      Enter the LDAP attribute whose value serves as the unique identifier for each account in the calendar database. The default is **davuniqueid**.

      > **Caution:**
      >
      > Do not use the value of **nsUniqueId** in a production deployment. See "Changing the User Unique Identifier" for more information about choosing a production-ready value.

    - User/Group default domain (The default value results from when you run the **comms_dssetup.pl** script against the Directory Server.)

      Enter the domain name for the LDAP users in the deployment.

- Default organization DN (The default value results from when you run the **comms_dssetup.pl** script against the Directory Server.)

  Enter the organization DN under which all users and groups that belong to the default domain are located in the LDAP tree.
- Calendar Server administrator user
- Calendar Server administrator user password

  If other messages appear when validation failed, you are prompted to re-enter all the Directory Server settings in this step.

11. Enter the Notification Mail Server Configuration Details.
    - Mail Server Host Name
    - Mail Server Port Number
12. When prompted whether to proceed with configuring Calendar Server, answer **Y**.

    The configurator displays messages indicating its actions and progress. The last messages indicate the location where the installation log file was written.
13. Restart the application server.

Running init-config Script in Silent Mode (Non-Interactive)

> **Note:**
>
> - If you use GlassFish Server, see "Calendar Server Configuration Scripts" for more information.
> - If you use WebLogic Server, obtain the correct **statefile** to run using **init-config** in a silent mode.
> - You should not use the **statefile** from the past release versions.

Perform the following step to obtain the correct **statefile** to run using **init-config** in a silent mode:

- Complete the interactive-mode **init-config -W** configuration.

  After the configuration is completed, a **statefile** is created in this location:

  ```
  CalendarServer_home/davserver/install/davserver-config-timestamp/saveState
  ```

  > **Note:**
  >
  > If you want to save the **statefile** in a different location, run **init-config** in an interactive mode as: **init-config -W -S** *customlocation_for_statefile*.

About the new parameter: From the Calendar Server 8.0.0.4.0 release onwards, a new parameter **appsrv.isweblogic** is created inside the statefile. Therefore, when Calendar Server is deployed on WebLogic Server, the **appsrv.isweblogic** parameter is set to **true**. If the **appsrv.isweblogic** parameter is set to **false** or the absence of this parameter indicates that Calendar Server is deployed on GlassFish Server.

If you want to re-run the Calendar Server configurator for WebLogic Server in a silent mode in the future, you can reuse this statefile as shown below:

```
init-config -s -f path_to_statefile_saved_during_freshconfig_usingWeblogic
```

Next Steps

After configuring Calendar Server, continue with the following chapters:

- Go to "Configuring Calendar Server with Multiple Hosts" if you have multiple hosts in your deployment.
- Follow the instructions in "Configuring the Calendar Server Document Store" to configure the document store.
- Follow the instructions in "Calendar Server Post-Installation Tasks" to perform post-installation tasks.

6 Configuring Calendar Server with Multiple Hosts

This chapter describes how to configure multiple Oracle Communications Calendar Server back-end hosts. For conceptual information, see Calendar Server Front-End and Back-End Components For information on performing an initial installation of Calendar Server, including back-end hosts, see "Installing Calendar Server".

About Installing and Configuring Multiple Calendar Server Back-End Hosts

A standard Calendar Server installation consists of a default back-end database that contains user data, and an iSchedule back-end database for iScheduling requests. Over time, you might want to add additional back-end user data bases to your initial deployment.

In the case of multiple Calendar Server front ends, configure each to use the same initial default database back end and iSchedule back end. Then, you add additional back ends to each front end. Only one iSchedule back end is needed for all front ends to share.

Each back-end database, including the iSchedule database, must have its own document store. In the case of multiple front ends, all document stores must be available to all front ends. You cannot make all document stores local to a front end in a multiple front-end deployment.

The high-level steps to configure multiple back-end Calendar Server hosts include:

1. Installing, configuring, and preparing the new database
2. Gathering the database host names, ports, and other database names
3. Installing all front-end hosts, if not already done
4. Configuring all front-end servers by using the **init-config** script, if not already done

   Running the **init-config** script creates the **config-backend** script for use in the next step. Information on any back-end host can be used for this step. If you have already run the **init-config** script, and you want to rename the default back-end host, see "Renaming the Default Calendar Server Back-End Host".
5. Running the **config-backend** script on each front-end host.

   > **Note:**
   >
   > If you use WebLogic Server, you cannot run this script. See "Installing and Configuring Multiple Calendar Server Back-End Hosts for WebLogic Server Manually" for more information.
6. Enabling connection pool validation on each front-end host.

> **Note:**
>
> This step is applicable only for GlassFish Server.

7. Restarting the application server on each front-end host.

Skip Steps 2 and 3 if you are adding new back-end hosts to an existing front-end installation.

Installing and Configuring Multiple Calendar Server Back-End Hosts for GlassFish Server

To install and configure multiple Calendar Server back-end hosts for GlassFish Server:

1. Install the database software on each back-end host. Choose one of the following:
   - Installing and Configuring MySQL Server
   - Installing and Creating an Oracle Database Instance for Calendar Server
2. Decide if you must create additional Oracle Database or MySQL Server back-end databases. If MySQL, continue with this step. If Oracle Database, skip to Step 3.

   > **Note:**
   >
   > If the Calendar Server software is not installed on the back-end host, copy the **config-mysql**, **config-oracle**, and **Util.pm** scripts from an installed Calendar Server host and adjust the following path to those scripts accordingly.

   - If this is first database on the host, set up the instance, and create the user and database by running the following command.

     ```
     Calendar_Home/tools/unsupported/bin/config-mysql -s -u -c
     ```

   - If there is already a database on the host, just create the calendar database by running the following command.

     ```
     Calendar_Home/tools/unsupported/bin/config-mysql -c
     ```

3. To create the Oracle database user and schema, see the following:
   - Preparing Oracle Database 11g Release 2 or Oracle Database 12c Not Pluggable (non-CDB)
   - Preparing Oracle Database 12c Container Database
4. On each front-end host, run the **config-backend** script.

   This script creates a JDBC connection pool and a JDBC resource on the GlassFish Server, and a **davserver** attributed back-end configuration.

   ```
   CalendarServer_home/sbin/config-backend
   ```

   - If current deployment is using MySQL, you are prompted for the following information:

     ```
     Remote database server host name
     Remote database server port
     Calendar db name on remote server
     Calendar db user name
     ```

```
Calendar db user password
Verifying the database input...
Database input is verified
Backend identifier for the remote db
Document store directory (leave blank if store is remote)
Document store host (leave blank if store is local)
Document store port (leave blank if store local)
Application Server admin user password
```

Make sure the value for **Calendar db name on remote server** is the one that you used for the **config-mysql -c** command.

- If current deployment is using Oracle Database, you are prompted for the following information:

```
Remote database server host name
Remote database server port
Oracle database service name on remote server
Calendar db user name
Calendar db user password
Verifying the database input...
Database input is verified
Backend identifier for the remote db
Document store directory (leave blank if store is remote)
Document store host (leave blank if store is local)
Document store port (leave blank if store local)
Application Server admin user password
```

Make sure the value for **Calendar db user name** is the one that you used for the **config-oracle -c** command.

5. Enter **Y** when prompted to perform the tasks for creating the JDBC connection pool and resource, and **davserver** back-end identifier.

   The system responds that the database back-end configuration is configured successfully.

6. On each front-end host, enable Connection Validation for the connection pools (both CalDav back-end and iSchedule pools) so that Calendar Server automatically reconnects to the back-end database if it goes down:

   a. In the GlassFish Server Administration Console, select **Resources**, then **JDBC**, then **Connection Pools**.

   b. Select the pool.

   c. Check the Required box for Connection Validation.

   d. Select **table** for the Validation Method.

   e. Enter **DUAL** for Table Name.

   f. Click **Save**.

   g. Click the **Advanced** tab.

   h. Enter **60** for Validate Atmost Once.

   i. Click **Save**.

   This configuration then issues the command **select count(*) from DUAL;** on every connection, at most one time every 60 seconds. (If the connection is not being used, it is not checked.)

> **Note:**
>
> Use these settings as a starting point and adjust where necessary. For example, if validation is not important, you can turn it off. Additionally, you might want to adjust the "Validate At Most Once" time duration or validate each time a connection is requested (by setting the value to 0). HA deployments might also use different values.

7. Restart GlassFish Server.
8. Provision accounts for a multiple back-end deployment.

   See "Provisioning Calendar Accounts in a Multiple Back-End Deployment".

> **Note:**
>
> If you use WebLogic Server, see "Installing and Configuring Multiple Calendar Server Back-End Hosts for WebLogic Server Manually" for more information.

Renaming the Default Calendar Server Back-End Host

The Calendar Server **init-config** script creates the JDBC connection pool and resource, and adds the information to the **davserver.properties** file, for the one back-end host specified during the front-end configuration. The JDBC resource used is **defaultbackend**.

If you must change this JDBC resource, to match other naming conventions, follow these steps:

1. On each front-end the application server, create a JDBC resource associated with the **caldavPool** connection Pool.

   For example, you might use **db1** as the resource name.
2. Save this change and then restart the application server.
3. Add the following two lines to each **davserver.properties** file.

   ```
   store.dav.db1.backendid=JDBC resource
   store.dav.db1.jndiname=jdbc/JDBC resource
   ```

   For example, if your resource name is **db1**, then you would add:

   ```
   store.dav.db1.backendid=db1
   store.dav.db1.jndiname=jdbc/db1
   ```

   The new resource *name* can be used in **davstore** attribute values.

> **Note:**
>
> Once your Calendar Server deployment is up and running, do not change the user back-end ID as defined by the **davStore** attribute.

Provisioning Calendar Accounts in a Multiple Back-End Deployment

For calendar accounts to know which back-end host they should connect to, you must provision accounts with the **davstore** attribute. The **davStore** attribute indicates the back-end host that stores a user's data if the deployment is configured for multiple back-end hosts. For more information, see the topic on Calendar Server and Directory Server integration in *Calendar Server Concepts*.

Examples: Creating Pools and Resources for Calendar Server Back-End Hosts Using GlassFish Server

The following examples apply only to Calendar Server 7 and Calendar Server 7 Update 1. Starting with Calendar Server 7 Update 2, this process is automated by the **config-backend** script.

This section contains the following topics:

- Using the GlassFish Server Administration Console to Create a Connection Pool
- Using the Command Line to Create a Connection Pool for Non-Default Back-End Hosts

Using the GlassFish Server Administration Console to Create a Connection Pool

This example uses the GlassFish Server Administration Console to create a connection pool **caldav1Pool** and JDBC resource **jbc/backend1**, and to enable Connection Validation.

1. Select **New** and enter the following information.
   - Name: **caldav1Pool**
   - Resource Type: **javax.sql.DataSource**
   - Database Vendor: **MySQL**
2. Click **Next**.
3. Set the following properties and be sure that the URL property is either not set or set correctly.

   You can also delete all the default properties and keep just the following six properties.
   - databaseName: **caldav1**
   - portNumber: **3306**
   - networkProtocol: **jdbc**
   - serverName: **localhost** (or your MySQL server host name)
   - user: **mysql**
   - password: **mysql**
4. Click **Save**.
5. Select the pool and use the **Ping** button to test the pool.

If ping was not successful, you can delete all the default properties and keep just the six properties previously mentioned. Then retry the ping.

6. Create **JDBC resource** for the connection pool.

   Select JDBC from Resources, then JDBC Resources.

7. Select **New** and enter the following information:
   - JDNI Name: **jdbc/backend1**
   - Pool Name: **caldav1Pool**
   - Status: check **Enabled**

8. Enable Connection Validation for the connection pool:
   a. In the GlassFish Server Administration Console, select **Resources**, then **JDBC**, then **Connection Pools**.
   b. Select **caldav1Pool**.
   c. Check the **Required** box for Connection Validation.
   d. Select **table** for the Validation Method.
   e. Enter **DUAL** for Table Name.
   f. Click **Save**.
   g. Click the **Advanced** tab.
   h. Enter **60** for Validate Atmost Once.
   i. Click **Save**.

Using the Command Line to Create a Connection Pool for Non-Default Back-End Hosts

This example uses the command-line interface to create a connection pool **caldav1Pool** and JDBC resource **jbc/backend1**.

```
GlassFish_home/bin/asadmin create-jdbc-connection-pool --user admin --
datasourceclassname com.mysql.jdbc.jdbc2.optional.MysqlDataSource --restype
javax.sql.DataSource --property
"DatabaseName=caldav1:serverName=mysqlhost:user=caldav:password=mysqlpass:portNumber=3306
:networkProtocol=jdbc" caldav1Pool

GlassFish_home/bin/asadmin create-jdbc-resource --user admin --connectionpoolid
caldav1Pool jdbc/backend1
```

Installing and Configuring Multiple Calendar Server Back-End Hosts for WebLogic Server Manually

To install and configure multiple Calendar Server back-end hosts for WebLogic Server manually, perform the following steps:

1. Install the database software on each back-end host. Choose one of the following:
   - Installing and Configuring MySQL Server
   - Installing and Creating an Oracle Database Instance for Calendar Server

2. Decide if you must create additional Oracle Database or MySQL Server back-end databases. If you have chosen MySQL, continue with this step. If you have chosen Oracle Database, skip to Step 3.

   > **Note:**
   >
   > If the Calendar Server software is not installed on the back-end host, copy the **config-mysql**, **config-oracle**, and **Util.pm** scripts from an installed Calendar Server host and adjust the following path to those scripts accordingly.

   - If this is the first database on the host, set up the instance, and create the user and database by running the following command:

     ```
     Calendar_Home/tools/unsupported/bin/config-mysql -s -u -c
     ```

   - If there is a database on the host already, create the calendar database by running the following command:

     ```
     Calendar_Home/tools/unsupported/bin/config-mysql -c
     ```

3. To create the Oracle database user and schema, see the following:
   - Preparing Oracle Database 11g Release 2 or Oracle Database 12c Not Pluggable (non-CDB)
   - Preparing Oracle Database 12c Container Database

4. Collect the following details specific to your new database backend setup.
   - *Frontend*: host1.us.example.com: Calendar Server deployed on WebLogic Server exists on this host.
   - *Database host*: host2.us.example.com: Existing backend database in production. Currently, frontend Calendar Server on host1 is configured to the backend database.
   - *New*: Additional database to be added: host3.us.example.com: Required database (MySQL database or Oracle database) is setup on this machine. This is associated with identifier **backend2**.

   Based on the above details, provide values for the following.

   If you use MySQL database:
   - Remote database server host name= *host3.us.example.com*
   - Remote database server port= *3306*
   - Calendar database name on remote server=*caldav*
   - Calendar database user name=*mysql*
   - Calendar database user password=*mysql*
   - Backend identifier for the remote database=*backend2*
   - Document store directory (leave blank if store is remote)= */var/opt/sun/comms/davserver/db/backend2*
   - Document store host (leave blank if store is local)
   - Document store port (leave blank if store is local)
   - Application Server admin user password=*adminpass* (Front-end WebLogic Server Admin User password)

If you use Oracle database:

- Remote database server host name=*host3.us.example.com*
- Remote database server port = *1521*
- Oracle database service name on remote server=*pdb2.us.example.com*
- Calendar database user name=*csadmin*
- Calendar database user password= *password*
- Backend identifier for the remote database= backend2
- Document store directory= */var/opt/sun/comms/davserver/db/backend2*

  > **Note:**
  >
  > Do not provide any value if the document store directory is remote.

- Document store host

  > **Note:**
  >
  > Do not provide any value if the document store host is local.

- Document store port

  > **Note:**
  >
  > Do not provide any value if the document store port is local.

- Application Server administrator user password= *adminpass*

  > **Note:**
  >
  > You should provide the front-end WebLogic Server Administration user password.

5. Create JDBC Data Source using WebLogic Server Administrator Console on your frontend machine.

   Log in to your front-end computer where Calendar Server is deployed on WebLogic Server. You must possess the WebLogic Server Administration credentials and have access to WebLogic Server Administration Console.

   Based on the values you have gathered in step 4, proceed to enter the details.

> **Note:**
>
> The following WebLogic Administration Console screen options are provided based on WebLogic Server 12.2.1.3.

6. Log in to WebLogic Server Administration Console on host1.us.example.com. For example, https://host1.us.example.com:7001/console
7. Click **Lock & Edit**.
8. Click your domain directory. For example, domain1.
9. Navigate to **Services** and then **Data Sources**.
10. Under the **Configuration** tab, click **New**.
11. Select **Generic Data Source**.
12. Enter the following details:
    - name: backend2
    - scope: global
    - JNDI Name: jdbc/backend2
    - If your database type is MySQL, select Database type: MySQL
      a. Click **Next**.
      b. Database Driver: Select MySQL's Driver (Type 4) Versions: using com.mysql.jdbc.jdbc2.optional.MySqlDataSource
      c. Click **Next**.

         You can leave the default values shown under the Transaction options.
      d. Click **Next**.
      e. Database Name: caldav.
      f. Hostname=host3.us.example.com
      g. port = 3306
      h. Database User Name=mysql
      i. password=mysql
      j. confirm password=mysql
      k. Click **Next**.
      l. Ensure that the following is shown in URL=jdbc:mysql://host3.us.example.com:3306/caldav
      m. Under Properties, (properties to pass to the JDBC driver when creating database connections), you can have the following: user=mysql, databaseName=caldav, characterEncoding=UTF8
      n. Test table Name: SQL SELECT 1
      o. Click **Test Configuration**.

         The result must be **Connection test succeeded**.
      p. Click **Next**.
    - If your database type is Oracle, select Database type: Oracle.

a. Click **Next**.
b. Database Driver: Select ***Oracle's Driver (Thin) for Service connections; Versions:Any**.
c. Click **Next**.
d. You can leave the default values shown under Transaction options.
e. Click **Next**.
f. Database Name: pdb2.in.example.com
g. Hostname=host3.us.example.com
h. port=1521
i. Database User Name=csadmin
j. password=password
k. confirm password=password
l. Click **Next**.
m. Ensure the URL is shown correctly as: URL=jdbc:oracle:thin:@//host3.us.example.com:1521/pdb2.in.example.com
n. Test table Name: SQL SELECT 1 FROM DUAL
o. Click **Test Configuration**.

   The result must be **Connection test succeeded**.
p. Click **Next**.

13. Select the target server.

    > **Note:**
    >
    > The target server must be your Managed Server. For example, server1. The selected target server must be the same target server that you had selected during the init-config setup on the front-end machine.

14. Click **Finish**.

    You can see the newly created backend2 in the list of **Data Sources** table.
15. Click **Activate Changes**.
16. Restart WebLogic Server.

    Ensure that WebLogic Admin or Managed Server log file does not contain any errors.
17. Run **davadmin backend create** where a new resource is created. For example, *backend2*.
18. Log in to your front-end host where the Calendar Server is set up. For example, *host1.us.example.com*.
19. Navigate to the *CalendarServer_home*/sbin directory where *davadmin* CLI tool is residing. For example, /**opt/sun/comms/davserver/sbin**/*davadmin*.
20. Run the *davadmin* command:

    ```
    /opt/sun/comms/davserver/sbin/davadmin backend create -u weblogic_adminuser -n
    newbackend_identifier -j newbackend_JNDIName -d  local_documentstorepath
    ```

For example,

```
/opt/sun/comms/davserver/sbin/davadmin  backend create -u weblogic -n  backend2 -j
"jdbc/backend2" -d  "/var/opt/sun/comms/davserver/db/backend2"
```

21. Run the following CLI command and verify the created backend details by checking the store.dav.xx parameters that are listed:

    /opt/sun/comms/davserver/sbin/*davadmin config* **list**

    When you run the above command, the list must contain the following:

    store.dav.backend2.jndiname=jdbc/backend2

    store.dav.backend2.dbdir=/var/opt/sun/comms/davserver/db/backend2

    store.dav.backend2.attachstorehost=

    store.dav.backend2.attachstoreport=8008

    store.dav.backend2.backendid=backend2

    store.dav.backend2.purgedelay=2592000

22. Provision accounts for a multiple back-end deployment.

    See "Provisioning Calendar Accounts in a Multiple Back-End Deployment".

User Login in a Multiple Back-End Host Deployment

When you deploy Calendar Server in a multiple back-end host configuration, the information flow for a user login is as follows:

1. The user logs in to Calendar Server.
2. The user's **davstore** LDAP attribute is read, indicating the back-end database with which the user is associated.
3. The **davstore** attribute is mapped to one of the **store.dav.***xx***.backendid** values in the Calendar Server **davserver.properties** file.
4. The corresponding JNDI name (**store.dav.***xx***.jndiname**) is obtained.
5. This points to a JDBC Resource associated with the back-end database.
6. The JDBC Resource points one of the JDBC Connection pools.
7. The user gets a Calendar Server database connection.

For example, assume that **user1** has a **davstore** attribute set to **backend1**. This would be mapped to the following **backendid** values:

```
store.dav.backend1.backendid=backend1
store.dav.backend1.jndiname=jdbc/backend1
```

The JNDI name would be obtained from:

```
store.dav.backend1.jndiname=jdbc/backend1
```

This, in turn, resolves to the following JDBC Resource:

```
Name=jdbc/backend1
ConnectionPool=caldav1Pool
```

Next, the following JDBC Connection pool is obtained:

```
ConnectionPool=caldav1Pool
```

```
Name=caldav1Pool
DB=jdbc:mysql://dbhost.example.com:3306/caldav1
```

Finally, **user1** is given a connection to the back-end host with the **caldav1** database.

7

Configuring the Calendar Server Document Store

This chapter describes how to configure the Oracle Communications Calendar Server document store.

About Configuring the Document Store

The Calendar Server document store is used to store and retrieve *large* data, such as calendar events and todos with large attachments. You must configure one document store per each Calendar Server back-end database. That is, configure one document store for the calendar store (the MySQL Server database or Oracle Database), and one document store for the iSchedule database.

The document store can be either local or remote, or, if you use Oracle Database, within the database itself. When the document store is local, Calendar Server accesses the file systems directly. When the document store is remote, Calendar Server accesses the documents by using either HTTP or HTTPS. If you use Oracle Database, you can configure it to contain both the calendar data and the data that otherwise would need to be stored in the separate document store. That is, you would use a single, *large* Oracle database for both calendar data and the document store.

Configuring Oracle Database for Both Calendar Data and Large Data

In the case of Oracle Database, you can configure it to contain both the calendar data and the data that would otherwise need to be stored in the separate document store. This type of configuration is not possible with a MySQL database. If you are using MySQL Server, refer instead to "Configuring a Local Document Store" and "Configuring a Remote Document Store" to set up a local or remote document store.

To configure Oracle Database for both calendar data and large data:

1. For each back end, set the **store.dav.***backend-name***.attachstorehost** parameter to a value of **dbdocstore**. For example:

   ```
   cd CalendarServer_home/sbin
   davadmin config modify -o store.dav.defaultbackend.attachstorehost -v dbdocstore
   ```

2. Restart Calendar Server by restarting the application server.

Configuring a Local Document Store

You can only configure a local document store when your deployment consists of a single Calendar Server front-end host. Do not configure a local document store on a front-end host if your deployment consists of multiple front-end hosts. Calendar Server requires that each front-end host needs access to each back-end host's database and document store pair.

To configure a local document store:

1. Log in to the local document store host.
2. Either use the default document store directory (**/var/opt/sun/comms/davserver/db/**), or create a data directory for the document store.

   ```
   mkdir path_to_docstore
   ```

3. Set the directory access permissions to be readable and writable by the document store user only:

   ```
   chmod 700 path_to_docstore
   ```

4. Configure Calendar Server to use the document store directory:

   ```
   davadmin config modify -u admin -o store.dav.defaultbackend.dbdir -v path_to_docstore
   ```

5. Restart Calendar Server by restarting the application server.

Configuring a Remote Document Store

Prerequisite: Install the Calendar Server software on the remote host. For more information, see "Installing Calendar Server".

To configure a remote document store:

1. Log in to the remote document store host.
2. Either use the default document store directory (**/var/opt/sun/comms/davserver/db/**), or create a data directory for the document store.

   ```
   mkdir path_to_docstore
   ```

3. Set the directory access permissions to be readable and writable by the document store user only:

   ```
   chmod 700 path_to_docstore
   ```

4. Run the **configure-as** script, which is located in the document store **sbin** directory:

   ```
   CalendarServer_home/sbin/configure-as
   ```

5. Change the **store.dav.***backend***.attachstorehost** and **store.dav.***backend***.attachstoreport** parameters on each Calendar Server front-end host to specify the host name and port number of the document store hosts for the back-end database.

   For example:

   ```
   davadmin config modify -u admin -o store.dav.defaultbackend.attachstorehost -v
   ahost.example.com

   davadmin config modify -u admin -o store.dav.defaultbackend.attachstoreport -v 8008
   ```

   Each back-end database and iSchedule database have their own document store. See the **davadmin** command documentation in *Calendar Server System Administrator's Guide* for more information about how to change the **store.dav.***backend***.attachstorehost** and **store.dav.***backend***.attachstoreport** parameters.

6. Restart Calendar Server by restarting the application server.

If you must start or stop the document store, the commands are:

```
CalendarServer_home/sbin/start-as
CalendarServer_home/sbin/stop-as
```

Configuring the Remote Document Store to Run as a Non-Root User

This procedure assumes that you have already configured the remote document store to run as **root**, and are now changing it to run as a non-root user.

1. On the remote document store host, make sure that the document store is stopped.

   ```
   CalendarServer_home/sbin/stop-as
   ```

2. On the remote document store host, run the **chown** command to change the user and group ownership of the files in the **davserver** configuration and data directory (default is **/var/opt/sun/comms/davserver**), and for the **start-as** and **stop-as** commands.

   ```
   chown -R non-root_user:group /var/opt/sun/comms/davserver
   chown non-root_user:group CalendarServer_home/sbin/start-as
   chown non-root_user:group CalendarServer_home/sbin/stop-as
   ```

3. If you have changed the default directory in the **ashttpd.properties** file for the **store.datadir** parameter, you must also change the ownership of that directory to the non-root user.

   For example, if **store.datadir** is set to **/export/davstore/nab**, then run the following command:

   ```
   chown -R non-root_user:group /export/davstore/nab
   ```

4. Start the remote document store as the non-root user.

   ```
   su - non-root_user -c "CalendarServer_home/sbin/start-as"
   ```

Configuring Remote Document Store Authentication

You must configure password authentication of the connection between the remote document store server (which runs on the remote host where the store is located), and the document store client (which runs on every Calendar Server front end). The password must be known by both the document store client and the remote document store server. The password is stored in a password file (called a wallet) on each host where it is needed.

This procedure assumes you have already created the remote document store and that it is running.

1. On the local Calendar Server host, use the **davadmin passfile modify** command to set the document store password.

   For example:

   ```
   cd CalendarServer_home/sbin
   davadmin passfile modify
   Enter the Password File password:

   Do you want to set the app server admin user password (y/n)? [n] n

   Do you want to set the database password (y/n)? [n] n

   Do you want to set the migration server user password (y/n)? [n] n

   Do you want to set the document store password (y/n)? [n] y
   Enter the document store password:
   Reenter the document store password:

   Do you want to set the document store SSL passwords (y/n)? [n] n
   ```

Setting the document store password updates the **store.document.password** configuration parameter.

2. On the remote document store host, configure the document store.

   ```
   cd CalendarServer_home/sbin
   configure-as

   Do you want to set the document store password (y/n)? [n] y
   Enter the document store password: password
   Reenter the document store password: password

   Do you want to set the document store SSL passwords (y/n)? [n] y
   Enter the document store SSL keystore password: <Enter the same password that you
   used when creating the self-signed certificate, that is, the keystore password>
   Reenter the document store SSL keystore password: password

   Enter the document store SSL certificate password: <Enter the same password that you
   used when creating the self-signed certificate, that is, the keystore password>
   Reenter the document store SSL certificate password: password

   Please check the values in /var/opt/sun/comms/davserver/config/ashttpd.properties
   are correct before starting the server with start-as
   To stop the server, run stop-as
   Document Store server is now configured
   ```

   To change the document store password, run the **davadmin passfile modify -O** command.

   For example:

   ```
   cd CalendarServer_home/sbin
   davadmin passfile modify -O

   Enter the Password File password:

   Do you want to set the document store password (y/n)? [n]

   Do you want to set the document store SSL passwords (y/n)? [n] y
   ```

   > **Note:**
   >
   > After setting the initial password, if you must change the password, rerun the **davadmin passfile modify** command. If you change the password by using the **davadmin config modify** command instead, to modify the **store.document.password** configuration parameter, the document store password in the "davadmin" wallet may not be up-to-date.

Configuring Remote Document Store SSL

You can use SSL to secure the transmission of data between the Calendar Server host and the document store. Configuring SSL between Calendar Server and the document store consists of the following high-level steps:

1. Configure the document store to accept SSL connections.
2. Configure Calendar Server to connect to the document store over SSL.

For more information, see the topic on configuring SSL for the remote document store in *Calendar Server Security Guide*.

Configuring the Document Store for High Availability

You can configure Calendar Server for multiple document stores such that, when the current document store host fails, the back-end database host fails over to the next available document store host.

To configure the document store for high availability:

1. Set up a shared file system, or some kind of data replication, for the document store data.
2. Use the **store.dav.***backendid***.attachstorehost** configuration parameter to specify a comma-separated list of document store hosts.

   For example:

   ```
   davadmin config modify -o store.dav.backendid.attachstorehost -v
   "ahost1.example.com,ahost2.example.com"
   ```

   When the current document store host fails, the back-end host fails over to the next document store host on the list.
3. Always keep the data on the document stores synchronized.

Migrating the Document Store

If, after installing the document store, you must migrate it to another location, you reconfigure the store location in the **ashttpd.properties** file, then manually move your existing documents in the **astore** directory. You also manually move the files in the **config** and **logs** directories to the new location before restarting the document store server and the front-end Calendar Server hosts. See the topic on document store configuration parameters in *Calendar Server System Administrator's Guide* for more information about the **ashttpd.properties** file.

To migrate the document store:

1. Stop Calendar Server by stopping the application server.
2. If the document store is remote, stop it as well:

   ```
   CalendarServer_home/sbin/stop-as
   ```
3. Manually move the existing documents from the old path to the new location.
4. Use one of the following methods to reconfigure the document store server, depending on whether it is local or remote:
   - Remote store: Reconfigure the **store.datadir** parameter in the **ashttpd.properties** file.

     See the topic on document store configuration parameters in *Calendar Server System Administrator's Guide* for more information about these parameters.
   - Local store: Reconfigure the **store.dav.defaultbackend.dbdir** parameter.
5. If the document store is remote, restart it:

   ```
   CalendarServer_home/sbin/start-as
   ```
6. Restart Calendar Server by restarting the application server.

Note:

If you use WebLogic Server, you must install Java version 8. For more information, see "Calendar Server Pre-Installation Tasks" and "Configuring Java for Calendar Server".

8

Calendar Server Post-Installation Tasks

This chapter provides instructions for Oracle Communications Calendar Server post-installation tasks.

Changing the User Unique Identifier

Calendar Server requires a unique identifier in the form of an LDAP attribute whose value is used to map each user account to a unique account in the database. The current default and recommended attribute, **davuniqueid**, prevents a potential serious issue with using **nsUniqueId**. If you use **nsUniqueId** and the LDAP entry for a user, group, or resource is deleted and recreated in LDAP, the new entry would receive a different **nsUniqueId** value from the Directory Server, causing a disconnect from the existing account in the calendar database. As a result, recreated users cannot access their existing calendars.

To change the unique identifier:

Run the **davadmin config** command to modify the **davcore.uriinfo.permanentuniqueid** configuration parameter. This parameter specifies the unique valued LDAP attribute present in the LDAP entry of all subjects (users, groups, and resources).

See the topic on command-line utilities in *Calendar Server System Administrator's Guide* for more information about the **davadmin** command.

> **Caution:**
>
> Changing this option after any user data is created in the database leads to data loss.

Calendar Server performs searches on the index you chose to use for **davcore.uriinfo.permanentuniqueid**. The installation process automatically creates the Directory Server index for **davuniqueid**. If you did not choose to use the default value of **davuniqueid** for **davcore.uriinfo.permanentuniqueid**, you must index the chosen attribute for presence and equality (**[pres.eq]**) in Directory Server. For more information about working with Directory Server indexes, refer to the Directory Server documentation.

Add the attribute to the list of LDAP attributes fetched by Calendar Server by running the **davadmin config modify** command to change the **davcore.uriinfo.subjectattributes** configuration parameter. Make sure to add on to the existing list and pass the entire value when doing the modification.

Configuring Virus Scanning

To enhance security within your installation, you can configure Calendar Server to scan attachments for viruses. Calendar Server virus scanning can examine attachments in a *real-time* mode to test and optionally reject incoming infected data. You can also choose to scan and optionally delete infected existing data *on-demand*.

To enable Calendar Server for virus scanning, see the topic on configuring and administering virus scanning in *Calendar Server System Administrator's Guide*.

Adding LDAP Access Control for Calendar Server Features

This section provides sample ACIs that show the attributes that Calendar Server needs for granting end users permission to search the LDAP directory for other users and resources whose calendars they can subscribe to. Tailor these samples to your individual site's security needs.

You add an ACI to the user/group suffix in Directory Server by using the **ldapmodify** tool. For more information on creating ACIs, see the topic on creating, viewing, and modifying ACIs in *Oracle Directory Server Enterprise Edition Administration Guide 11g Release 1 (11.1.1.5.0)*.

Sample ACIs:

- The following sample ACI enables users to search for all other users and resources in the same domain for all domains hosted in the Directory Server, assuming a suffix of `o=isp`.

  ```
  dn: o=isp
  changetype: modify
  add: aci
  aci: (target="ldap:///($dn),o=isp")
   (targetattr!="userPassword")
   (targetfilter=(|(objectClass=icscalendaruser)(objectclass=icscalendarresource)))
   (version 3.0; acl "CS User search access to all users and resources in own domain -
  product=davserver,class=install,num=3,version=1";
   allow (read,search)
   userdn="ldap:///[$dn],o=isp??sub?(objectclass=icscalendaruser)";)
  ```

- To control domain access at a finer level, add individual ACIs instead of using the **$dn** macro. For example, to allow search for only one domain, set the following ACI on the domain organization entry.

  ```
  dn: domainA.orgdn
  changetype: modify
  add: aci
  aci: (targetattr!="userPassword")
   (targetfilter=(|(objectClass=icscalendaruser)(objectclass=icscalendarresource)))
   (version 3.0; acl "CS User search access to all users and resources in domain A -
  product=davserver,class=install,num=3,version=1";
   allow (read,search)
   userdn="ldap:///domainA.orgdn??sub?(objectclass=icscalendaruser)";)
  ```

  Replace *domainA.orgdn* with your organization DN.

When adding ACIs to enable users to search other domains, that is, cross-domain searches, keep the following in mind:

- For domain A users to be able to search for users in domain B, you would add an ACI on domain B's node that allows the search from users in domain A. For example, to enable users in **example.com** to search users in **varrius.com**, add the following ACI to the domain entry for **varrius.com domain o=varrius.com,o=isp** by using the **ldapmodify** command:

  ```
  dn: o=varrius.com,o=isp
  changetype: modify
  add: aci
  aci: (targetattr!="userPassword")
   (targetfilter=(|(objectClass=icscalendaruser)(objectclass=icscalendarresource)))
   (version 3.0; acl "example.com users access in varrius.com -
  ```

```
product=davserver,class=install,num=3,version=1";
 allow (read,search)
 userdn="ldap:///o=example.com,o=isp??sub?(objectclass=icscalendaruser)";)
```

- You might also need to set domain Access Control Lists (ACLs) to control calendar operations that span multiple domains. Calendar Server combines domain ACLs with the calendar and scheduling ACLs to grant or deny levels of access to these operations. All operations within a single domain rely strictly on the calendar and scheduling ACLs. For more information, see the topic on managing domain access controls chapter in *Calendar Server Security Guide.*

Using the iSchedule Channel to Handle iMIP Messages

Messaging Server is capable of posting a calendar event received in an iCalendar Message-Based Interoperability Protocol (iMIP) message to Calendar Server by using the iSchedule protocol. This capability enables "internal" users to automatically process calendar invitations from "external" users.

To enable this interoperability between calendaring systems, you configure a Messaging Server "iSchedule" channel to process the iMIP messages. For more information, see the topic on using the iSchedule channel to handle iMIP messages in *Messaging Server System Administrator's Guide*.

Next Steps

Verify your Calendar Sever installation by configuring a client to connect to the server. See the topic on configuring CalDAV clients in *Calendar Server System Administrator's Guide*.

9 Upgrading Calendar Server

This chapter explains how to upgrade your existing system to the latest release of Oracle Communications Calendar Server. If you are running Calendar Server 6 (also known as Sun Java System Calendar Server 6), you must migrate your data to Oracle Communications Calendar Server. See "Migrating to Calendar Server" for more information.

About Upgrading Calendar Server

If your deployment uses document stores, upgrading requires you to also upgrade those document stores at the same time, as described in this chapter.

When performing an upgrade of Calendar Server, you must upgrade all front-end hosts and remote document store hosts (if applicable) to the same version. That is, you cannot have a deployment of mixed Calendar Server versions. Calendar Server upgrades may introduce back-end database schema changes, or the front-end hosts may access the database in a different way. Thus, all hosts must be at the same version.

> **Note:**
>
> If you are already running Calendar Server 8.0.0.4.0 with WebLogic server in front-end and want to upgrade to Calendar Server 8.0.0.5.0, run **commpkg upgrade** directly.
>
> For more information, see "Migrating to Calendar Server".

About Calendar Server Database Upgrades

Each Calendar Server patch or release might require an upgrade to the database schema or structure. For example, a database table might be altered, or data in the database might be changed. This type of upgrade differs from a MySQL Server or Oracle Database version upgrade. If a Calendar Server database schema or structure upgrade is required, allow additional time for the overall Calendar Server upgrade, as the database upgrade can take a while, especially if the database is large.

Topics in this section:

- Database Schema and Structure Version
- Calendar Server Start Up and Automatic Database Upgrade
- Database Schema Upgrade Patches and Releases
- Database Performance During Upgrade
- About Upgrading the Database Schema
- Preupgrade Functions

Database Schema and Structure Version

Each database schema and structure has a Calendar Server back-end version number recorded in the database. Each Calendar Server version only runs with a single database schema version.

You cannot reverse or downgrade a database schema version to a previous version. Thus, always ensure that you take a full database backup prior to upgrading. Once a database is upgraded, if you attempt to revert to a previous Calendar Server version, the previous Calendar Server version does not recognize the newly upgraded database. However, not all Calendar Server versions involve a database schema upgrade. When this is the case, you can revert to a previous Calendar Server version without invalidating the databases. Use Table 9-1 to understand which Calendar Server versions use the same database schema.

Calendar Server Start Up and Automatic Database Upgrade

When Calendar Server starts up in the application server, it connects to each back-end database for which it is configured, one at a time, and attempts to open the database for normal operation.

During this initial opening of each database, Calendar Server checks the database schema version. Calendar Server has the option to either automatically upgrade the database immediately, or issue a logging message in the **errors.0** log file explaining the database cannot be opened and needs to be upgraded.

> **Caution:**
>
> Never interrupt or stop either Calendar Server or the application server when the database is being upgraded. If you do, you must restore the database from backup.

Calendar Server logs messages in the **errors.0 log** file when the database was opened but not that it was being upgraded. If the database has a large amount of data, the upgrade can take a long time and appear to resemble a database hang. See "Database Schema Upgrade Patches and Releases" for more information on Calendar Server versions that require a database upgrade.

Monitoring the MySQL Server Database Upgrade

To monitor the database during an upgrade process on MySQL Server, use the following command:

```
mysql> SHOW FULL PROCESSLIST;
```

This command displays database upgrade aspects such as an **ALTER TABLE** command.

Database Schema Upgrade Patches and Releases

You cannot reverse a database schema and structure upgrade once it has been performed. This is why, when performing a Calendar Server upgrade, you must make a backup before starting the upgrade process. Once a database is upgraded, it no longer works with a previous Calendar Server version. Use Table 9-1 to understand the Calendar Server database changes that have occurred.

Table 9-1 Database Schema Version Changes for Calendar Server Beginning with Release 7 Update 2

Calendar Server Release	Database Change	Notes
8.0.0.1	Automatic upgrade to version 10.	Fairly intensive, depending on the size of existing data.
8.0	No database changes.	NA
7.0.5.17.0	Automatic upgrade to version 9.	No schema upgrade performance issues.
7.0.4.16.0	No database changes.	NA
7.0.4.15.0	No database changes.	NA
7.0.4.14.0	Automatic Upgrade to Version 8. (MySQL Server only) The charset and collation of most tables and fields is changed to **utf8mb4**. • 8 **ALTER TABLE** commands are run. Column added to CalProp. • 1 **ALTER TABLE** command is run.	Fairly intensive for MySQL Server.
7.0.3.8.0	Automatic Upgrade to Version 7. Full table scan fixes missing values in **Resources.resourcetype = 'CAL_RESOURCE';**.	NA
7.0.2.4.0	Automatic Upgrade to Version 6. • Rename table **Resource** to **Resources**. • **ALTER TABLE** command run on **CalProp** to change **supported_calendar_component_set** to **supported_cal_component_set**. • Change table **ICalProp** to **InnoDB** (by dropping it and allowing it to repopulate).	NA

Database Performance During Upgrade

How a database performs during an upgrade depends on data size and hardware. Some database operations, such as copying the entire table, are intensive operations and can take a long time to complete. When a database has 4 million rows or 40 gigabytes of data, study the database performance to better understand potential upgrade times.

MySQL Server and Oracle Database have relevant buffer cache settings that can increase performance. Additionally, if the database can fit in the cache completely with extra space, then operations might run faster.

If the database and temporary table operations exceed the database cache, then the disk IO throughput becomes very relevant. Disk IO is also significant when there is plenty of cache because the cache must be loaded at times.

MySQL Server also has memory and cache parameters such as **innodb_buffer_pool_size**, while Oracle Database has parameters such as **sga_target**. Refer to the MySQL Server and Oracle Database documentation for more information on performance tuning.

About Upgrading the Database Schema

In general, upgrading the database schema should not take a long time. However, there are some exceptions. See Table 9-1 for more information. Additionally, as explained in Database

Performance During Upgrade other factors can impact the time required to upgrade the database schema, and thus the overall Calendar Server upgrade time.

To increase database upgrade functionality and flexibility, you can use the **davadmin db schema_fullupgrade** and **davadmin db schema_preupgrade** commands. Table 9-2 describes these two commands.

Table 9-2 Comparison of schema_fullupgrade and schema_preupgrade

Description of schema_fullupgrade	Description of schema_preupgrade
The **schema_fullupgrade** command fully upgrades the back-end database schema. The **schema_fullupgrade** command performs all upgrade functions on the database, including upgrading the presence and all database tables for charset. The **schema_fullupgrade** command is the same as starting a new Calendar Server instance, which upgrades the database. Use this command to upgrade multiple back-end databases in parallel instead of one at a time, as happens during the normal Calendar Server upgrade process. For example, if your deployment consists of four back-end databases, instead of going through the normal upgrade process where upgrading a front-end host then initiates the back-end database upgrade, one at a time, you can upgrade the front-end hosts then upgrade all back-end databases in parallel by using the **schema_fullupgrade** option. After running the **schema_fullupgrade** command and the database schema upgrade completes, you cannot revert to the previous schema. In addition, Calendar Server front-end hosts that run a prior software version are not able to communicate with the database that has had its schema upgraded.	The **schema_preupgrade** command enables you to perform online database DDL changes, as well as a partial offline upgrade, as long as the release contains database schema updates. (Not all releases update the database schema, so **schema_preupgrade** might not apply.) For example, if your deployment contains a large amount of data, you can save time by pre-upgrading the database schema in advance of the formal Calendar Server software upgrade. The **schema_preupgrade** command does not change the database schema version, and therefore, Calendar front-end hosts continue to be able to communicate with the pre-upgraded database without having to be upgraded themselves. You specify the function to upgrade by using with the **-z** *preupgradefunction* option. See Table 9-3 for a list of functions by release. For example, if a Calendar Server upgrade adds a new column to the database, you can run the **schema_preupgrade** command to add that column to the current database. The Calendar Server front-end hosts, which have not yet been upgraded themselves, continue to have the capability to run with the pre-upgraded database version. When you later perform the formal Calendar Server software upgrade, the schema update has already been done, and so you save time on completing the overall upgrade process.

See the **davadmin db schema_fullupgrade** and **davadmin db schema_preupgrade** commands in "Calendar Server Command-Line Utilities" in *Calendar Server System Administrator's Guide* for more information.

Preupgrade Functions

The **davadmin db schema_preupgrade -z** *preupgradefunction* command enables you to specify which pre-upgrade function(s) to run on the database. Table 9-3 describes the available functions by release.

Table 9-3 schema_preupgrade Functions

Database Version	Function	Description
Version 8	**v8presence**	Upgrades for presence for both MySQL Server and Oracle Database
Version 8	**v8charsetresources**	MySQL only.
Version 8	**v8charsetcalprop**	MySQL only.

Table 9-3 (Cont.) schema_preupgrade Functions

Database Version	Function	Description
Version 8	**v8charseticalprop**	MySQL only.
Version 8	**v8charsetxprop**	MySQL only.
Version 8	**v8charsetcollection**	MySQL only.
Version 8	**v8charsetowner**	MySQL only.
Version 8	**v8charsetcontent**	MySQL only.
Version 8	**v8charsetattachment**	MySQL only.
Version 8	**v8charset**	MySQL only. Upgrades all tables for charset

Calendar Server Upgrade Dependencies

Table 9-4 describes the upgrade dependencies on Java, MySQL Server, application server, and other components for Calendar Server.

Table 9-4 Calendar Server Upgrade Dependencies

Upgrade Dependency	Introduced in Which Calendar Server Version?
Upgrading to Solaris 11.	Calendar Server 8 is supported on Solaris 11 only. If you are running Calendar Server 7 on Solaris 10, you must upgrade to Solaris 11 before upgrading to Calendar Server 8.
Deciding if you must change the LDAP attribute used as the unique identifier for your Calendar Server deployment.	Starting with Calendar Server 7 Update 3, if you initially deployed Calendar Server 7 Update 2 and prior to use the **nsUniqueId** attribute as your unique identifier, you should switch to using the **davUniqueId** attribute.
Installing Java 7 on all Calendar Server hosts, including front ends and document store hosts.	Starting with version 7.0.4.15.0, Calendar Server requires Java 7.
Installing and running the **comm_dssetup** script against your Directory Server hosts.	Starting with version 7.0.4.14.0, Calendar Server requires that at least **comm_dssetup.pl** 6.4.0.25.0 be run against your Directory Servers.
Upgrading MySQL to at least 5.5 (5.5.8 or greater).	Starting with Calendar Server 7 Update 2, you must use at least MySQL 5.5.8. Note that starting with Calendar Server 7.0.5, MySQL 5.6 is supported.
Making database charset change for installations using MySQL as the back-end database.	Calendar Server 7.0.4.14.0 and greater requires the MySQL Server default character set be set to **utf8mb4**. This is to support 4-byte Unicode, which the older MySQL UTF-8 character set does not support.
Creating the iSchedule database and granting permission to **caldav** user to access the iSchedule database.	If you are upgrading from Calendar Server 7 or Calendar Server Update 1, you must manually create the iSchedule database.
Installing GlassFish Server 3.1.2	Calendar Server 7.0.4.14.0 and greater requires that you use GlassFish Server 3.1.2 as its web container.
Running **davadmin account upgrade**.	Starting with version 7.0.4.14.0, Calendar Server requires that you run this command to set the next presence triggers for all existing events in the future.

The next section describes the actual upgrade procedures and indicates where the preceding dependencies apply.

Upgrading to Calendar Server 8.0

This section describes how to upgrade to Calendar Server 8.0. Depending on your current Calendar Server version, some steps are not required.

Prerequisites for Upgrading Calendar Server

Before upgrading Calendar Server, you must:

1. Upgrade to Solaris 11, if you are currently running Calendar Server 7 on Solaris 10.
2. Download the necessary software.
   a. Download the Calendar Server software and Directory Server setup script, **comm_dssetup**, either as part of a media pack, or as a patch.
      - Download the media pack from the Oracle software delivery website.
      - Download a Calendar Server patch or Directory Server setup patch from My Oracle Support.

   b. If upgrading from Calendar Server 7 Update 1 or prior version: download MySQL Server 5.5.8 or later from My Oracle Support.

      The MySQL download is available in both PKG and TAR versions. This documentation uses the PKG version to show installation and configuration, so in most cases, choosing the PKG version makes sense.

      Starting with version 7 Update 2, Calendar Server requires that you upgrade to MySQL Server 5.5.8 or any later 5.5.x version. As of Calendar Server 7.0.5, MySQL 5.6 is supported. See "Upgrading the Calendar Server Back-End Database".

      If you are running a 5.5.x version prior to 5.5.32, you can choose to upgrade but it is not necessary.

   c. Download the GlassFish Server 3.1.2 software from My Oracle Support.

   d. Download the Java 7 software from Java SE Downloads at:

      `http://www.oracle.com/technetwork/java/javase/downloads/index.html`

3. Place the software on the appropriate hosts.
   a. On Calendar Server front-end hosts, and, if applicable, remote document store hosts, create a temporary directory (for example, **/temp/CS8**), and extract the Calendar Server media pack zip file or Calendar Server patch zip file in this directory.

   b. If you are upgrading MySQL Server, on the Calendar Server back-end hosts, create a temporary directory (for example, **/temp/MySQL**), and extract the MySQL Server package in this directory.

   c. If you are upgrading the application server, on the Calendar Server front-end hosts, create a temporary directory (for example, **/temp/GF3**), and extract the application server zip file in this directory.

   d. On Calendar Server front-end hosts, and, if applicable, remote document store hosts, place the Java 7 software.

Backing Up the Calendar Server Database

The purpose of backing up the database is in case you must restore the previous Calendar Server version and database. However, database backups are not compatible across releases because backups include database internals specific to the version.

For more information, see the topic on backing up and restoring files and data in *Calendar Server System Administrator's Guide*.

Upgrading the Database Schema

You may be able to upgrade parts of the database schema as a separate step, outside of and in advance of, the formal Calendar Server upgrade process, if pre-upgrade functions exist for the release. (Not all Calendar Server versions update the database schema, so this capability might not apply.) Before upgrading the database schema, ensure that you understand the implications as described in "About Upgrading the Database Schema". Also, review Table 9-1 to confirm if the Calendar Server version upgrades the database schema. See the topic on the **davadmin db schema_preupgrade** command in *Calendar Server System Administrator's Guide* for more information.

Installing Java 7 or 8

Calendar Server 7.0.4.15.0 and greater requires that you use Java 7. If you have previously upgraded to Calendar Server 7.0.4.15.0, you should have already performed this step.

> **Note:**
>
> In Calendar Server 8.0.0.4.0 the WebLogic container support is added and it requires Java 8. Therefore, ensure that you have performed the instructions specific to WebLogic Server provided in "Calendar Server Pre-Installation Tasks" and "Configuring Java for Calendar Server".

The 32-bit and 64-bit JDKs require manual installation. Install both JDKs, rather than the JRE, on your front-end hosts

> **Note:**
>
> If you changed the Java 6 **java.policy** file, then you must do the same to the Java 7 file. If you installed Java 6 certificates in the **cacerts** file, you must also install those certificates in Java 7. See the Java 7 documentation for more information on these items.

Preparing the Directory Server

If you have previously upgraded to Calendar Server 7.0.4.14.0, you should have already performed this step. Calendar Server 7.0.4.14.0 and greater requires that at least **comm_dssetup.pl** 6.4.0.25.0 be run against your Directory Server hosts. Before you can upgrade to Calendar Server 8.0, you must run the **comm_dssetup.pl** script (minimum version

6.4.0.25.0) on each Directory Server in your deployment. This version of **comm_dssetup.pl** applies the required Directory Server index on the **davUniqueId** attribute for Calendar Server.

To run **comm_dssetup.pl**:

1. On each Directory Server host, run **commpkg install** (or **commpkg upgrade** if **comm_dssetup.pl** exists already on the host), and select only Comms DSsetup 6.4.
2. On each Directory Server host, run the **comm_dssetup.pl** script to apply the schema updates.

   For example:

   ```
   /opt/sun/comms/dssetup/sbin/comm_dssetup.pl
   ```
3. Respond to the prompts.

   For more information, see "comm_dssetup.pl Reference".

Running commpkg upgrade

This section assumes that you have previously put the Calendar Server software onto each Calendar Server front-end host and remote document store host, if applicable. If instead you are performing an upgrade of the Calendar Server software by applying a patch, see the instructions in "Installing Patches" then return to this chapter to resume the upgrade process.

1. On the Calendar Server front-end hosts, run **commpkg upgrade** and select the Calendar Server component.

   For more information, see "upgrade Verb Syntax".

   If the upgrade detects a problem during the pre-upgrade phase, correct the problem and re-run the upgrade.
2. If you have remote document stores, continue with the next section.

Upgrading the Remote Document Store

When upgrading Calendar Server front-end hosts, you must also upgrade the remote document store hosts at the same time. These instructions describe both how to upgrade document stores that are not configured to use SSL, and how to add SSL if you choose. These instructions also assume that you have downloaded and extracted the Calendar Server software zip file or Calendar Server patch zip file onto the remote document store hosts.

Topics:

- Upgrading a Non-SSL Document Store
- Upgrading a Non-SSL Document Store to SSL

Upgrading a Non-SSL Document Store

To upgrade a document store that is not configured for SSL:

1. Perform the following steps on the remote document store host.
   a. Stop the document store server.

      ```
      cd CalendarServer_home/sbin
      stop-as
      ```

b. Upgrade the Calendar Server software by launching the installer and choosing Calendar Server. If you are applying a patch, see the instructions in "Installing Patches" then return to this chapter to resume the upgrade process.

```
commpkg upgrade
```

c. Configure the document store and set the password.

```
cd CalendarServer_home/sbin
configure-as

Do you want to set the document store password (y/n)? [n] y
Enter the document store password: password
Reenter the document store password: password

Do you want to set the document store SSL passwords (y/n)? n

Please check the values in /var/opt/sun/comms/davserver/config/ashttpd.properties
are correct before starting the server with start-as
To stop the server, run stop-as
```

d. Start the document store server.

```
cd CalendarServer_home/sbin
start-as
```

2. Perform the following steps on the Calendar Server front-end host after you have upgraded the Calendar Server software.

a. Set the same document store password that you used during the configuration on the document store host.

```
cd CalendarServer_home/sbin
davadmin passfile modify
Enter Admin password: password
Enter the Password File password: password

Do you want to set the app server admin user password (y/n)? [n]

Do you want to set the database password (y/n)? [n]

Do you want to set the migration server user password (y/n)? [n]

Do you want to set the document store password (y/n)? [n] y
Enter the document store password: <Enter the same password that you used when
running the configure-as command on the document store host.>
Reenter the document store password: password

Do you want to set the document store SSL passwords (y/n)? [n] n

Set new value for store.document.password.
```

b. Restart the application server.

Upgrading a Non-SSL Document Store to SSL

Starting with Calendar Server 7.0.4.14.0, you can configure the Secure Sockets Layer (SSL) protocol between the Calendar Server front ends and back ends, including the remote document store.

To upgrade a non-SSL document store to SSL:

1. Perform the following steps on the remote document store host.

   a. Stop the document store server.

      ```
      cd CalendarServer_home/sbin
      stop-as
      ```

   b. Upgrade the Calendar Server software by launching the installer and choosing Calendar Server. If you are applying a patch, see the instructions in "Installing Patches" then return to this chapter to resume the upgrade process.

      ```
      commpkg upgrade
      ```

2. Create a self-signed certificate.

   ```
   keytool -genkeypair -alias alias -keyalg RSA -validity 365 -keysize 2048 -keystore
   path/keystore_name.jks
   ```

3. Export the certificate and copy it to the Calendar Server host.

   ```
   keytool -export -alias alias -keystore path/keystore_name.jks -rfc -file path/
   certificate.cert
   ```

4. Configure the document store.

   ```
   cd CalendarServer_home/sbin
   configure-as

   Do you want to set the document store password (y/n)? [n] y
   Enter the document store password: password
   Reenter the document store password: password

   Do you want to set the document store SSL passwords (y/n)? [n] y
   Enter the document store SSL keystore password: <Enter the same password that you
   used when creating the self-signed certificate, that is, the keystore password>
   Reenter the document store SSL keystore password: password

   Enter the document store SSL certificate password: <Enter the same password that you
   used when creating the self-signed certificate, that is, the keystore password>
   Reenter the document store SSL certificate password: password

   Please check the values in /var/opt/sun/comms/davserver/config/ashttpd.properties
   are correct before starting the server with start-as
   To stop the server, run stop-as
   Document Store server is now configured
   ```

5. Add the following lines to the **ashttpd.properties** file.

   ```
   store.usessl=true
   store.sslkeystorepath=path/keystore_name.jks
   ```

6. Start the document store.

7. Perform the following steps on the Calendar Server front-end host after you have upgraded the Calendar Server software.

8. Import the certificate (the certificate that you copied from the document store host) into the TrustStore on the application server host where you have deployed Calendar Server.

   ```
   keytool -importcert -alias alias -file path/certificate.cert -keystore /opt/
   glassfish3/glassfish/domains/domain1/config/cacerts.jks
   ```

9. List the KeyStore to ensure that the alias exists.

   ```
   keytool -list -keystore /opt/glassfish3/glassfish/domains/domain1/config/cacerts.jks
   -alias alias
   ```

10. Set the same document store password that you set during the document store configuration.

```
davadmin passfile modify
Enter Admin password: password
Enter the Password File password: password

Do you want to set the app server admin user password (y/n)? [n]

Do you want to set the database password (y/n)? [n]

Do you want to set the migration server user password (y/n)? [n]

Do you want to set the document store password (y/n)? [n] y
Enter the document store password: <Enter the same password that you gave during
configure-as on document store host>
Reenter the document store password: password

Do you want to set the document store SSL passwords (y/n)? [n] n

Set new value for store.document.password. A server restart is required for this
change to take effect.
```

11. Set the **store.document.usessl** configuration parameter to **true**.

    ```
    davadmin config modify -o store.document.usessl -v true
    Enter Admin password: password
    Set new value for store.document.usessl. A server restart is required for this
    change to take effect.
    ```

12. Restart the application server.

Upgrading the Calendar Server Back-End Database

Ensure that you are running a supported database as described in "Required Software". If you are running MySQL Server 5.1, which was part of the Calendar Server 7 Update 1 version, you must upgrade to at least MySQL Server 5.5.8.

Refer to the appropriate MySQL Server and Oracle Database documentation for instructions about upgrading your database.

Before upgrading your database, always make a full backup, for example, by using the **davadmin** command:

```
davadmin db backup -k backup_file
```

Additionally, you might want to dump the Calendar Server back-end data in case a problem happens during the upgrade and you must reload your data. For example, on MySQL Server, run the following command:

```
mysqldump --user=caldav --password=password --all-databases > myDataDump.sql
```

Also, you must stop the Calendar Server front-end services by shutting down the application server domain before starting the database upgrade.

When upgrading MySQL Server, ensure that you make the following changes in the **/etc/my.cnf** file:

- Ensure that the **my.cnf** file includes the transaction isolation level additional configuration parameter:

  ```
  transaction-isolation = READ-COMMITTED
  ```

- If you use replication, use the following **ROW** binary logging, instead of the default **STATEMENT** logging. Add the following to the **my.cnf** file:

```
binlog-format=ROW
```

- If you use **log_long_format**, it is no longer supported in MySQL 5.5.8. Remove it from the **my.cnf** file, otherwise MySQL Server does not start.

Updating the Calendar Server Back-end Database Charset on MySQL

If you have previously upgraded to Calendar Server 7.0.4.14.0, you should have already changed the character set. Calendar Server 7.0.4.14.0 and greater requires the MySQL Server default character set be set to **utf8mb4**. This is to support 4-byte Unicode, which the older MySQL UTF-8 character set does not support.

To set the MySQL Server default character set:

1. Edit the **/etc/my.cnf** file.
2. Change the following line:

   ```
   character-set-server = utf8
   ```

   to

   ```
   character-set-server = utf8mb4
   ```

Creating the iSchedule Database

If you have previously upgraded from Calendar Server 7 or Calendar Server 7 Update 1, you should have already created the iSchedule database. Calendar Server 7 Update 1 and greater requires that the iSchedule database exist, even if you do not use it.

Depending on your database, run the appropriate command:

- MySQL Server: **config-mysql --ischedule**
- Oracle Database: **config-oracle --ischedule**

> **Note:**
>
> You cannot use the **config-oracle** script for an Oracle Database 12c container database (that is, one that uses a pluggable database). Instead, you must manually create the iSchedule database for an Oracle Database 12c container database. See "Preparing Oracle Database 12c Container Database" for more information on the manual steps to create the iSchedule database.

For more information, see "Running the config-mysql Script" or "Running the config-oracle Script".

Installing GlassFish Server 3

If you are previously upgraded to Calendar Server 7.0.4.14.0, you should have already performed this step. Starting with version 7.0.4.14.0, Calendar Server requires that you use GlassFish Server 3.1.2 as its web container. You do not need to upgrade your current GlassFish 2.1.1 version. Instead, you install GlassFish Server 3.1.2 on another location using either the existing port used by GlassFish Server 2.1.1 or a new port. Then, when you run the Calendar Server **init-config** command, you enter the new values for the GlassFish Server 3.1.2 installation.

1. See *Oracle GlassFish Server 3.1.2 Installation Guide* for instructions on installing GlassFish Server software.
2. After installing GlassFish Server 3.1.2, enable secure administration.

   The GlassFish Server secure administration (**secure admin**) feature enables you to secure all administrative communication between the server and administration clients such as the **asadmin utility** and the administration console.

   ```
   /opt/glassfish3/bin/asadmin enable-secure-admin
   You must restart all running servers for the change in secure admin to take effect.
   Command enable-secure-admin executed successfully.

   /opt/glassfish3/bin/asadmin restart-domain
   Successfully restarted the domain
   Command restart-domain executed successfully.
   ```

Setting Environment Variables

Upgrading to Java 7 requires that you make the following environment variable changes. If you have previously upgraded to Java 7, you should have already performed these steps.

1. Set the **JAVA_HOME** environment variable to Java 7 in the login profile of the application server runtime user, or create a symbolic link for **/usr/jdk/latest** to the Java 7 location.
2. Stop GlassFish Server.
3. Set *GlassFish_home*/**glassfish/config/asenv.conf** to the Java 7 location.
4. Restart GlassFish Server.

> **Note:**
>
> If you use WebLogic Server, you must install Java version 8. For more information, see "Installing Java".
>
> If you use WebLogic Server, you must install Java 8. You must also set **JAVA_HOME** and **PATH** environment variables accordingly to the JDK 1.8.0 latest update version.

Running the populate-davuniqueid Script

Ignore this step if you have already switched to using the **davUniqueId** attribute. If you initially deployed Calendar Server 7 Update 2 to use the **nsUniqueId** attribute as your unique identifier, you should switch to using the **davUniqueId** attribute, which was introduced in Calendar Server 7 Update 3. For more information on the problems with using **nsUniqueId**, see "Changing the User Unique Identifier". For information on the **davUniqueId** attribute, see *Communications Suite Schema Reference Guide*.

To run the **populate-davuniqueid** script to populate, in LDAP, calendar users, groups, and resources with the **davUniqueId** attribute:

1. You can run the script interactively or with supplied parameters. For more information, see "populate-davuniqueid Examples". Here is a sample **populate-davuniqueid** session:

   ```
   /opt/sun/comms/davserver/sbin/populate-davuniqueid
   Directory host: ds.example.com
   Directory user bind DN: admin
   Directory user bind password:
   ```

```
Directory search base: o=isp
Output to file: popdavunique-output
Please check the following information
Directory host: ds.example.com
Directory port: 389
Directory user bind DN: admin
Directory user bind password: as specified
Directory search base: o=isp
Directory search filter: (|(objectclass=icscalendaruser)
(objectclass=icscalendarresource)(objectclass=groupofuniquenames)
(objectclass=groupofurls)(objectclass=inetmailgroup))
Output to: popdavunique-output
Add daventity objectclass: no
Load output LDIF automatically: no
Do you want to continue (y/n)?:
```

2. Use the **davadmin** command to modify the **davcore.uriinfo.permanentuniqueid** configuration parameter to **davUniqueId**.

   You must do this, even though you set **davUniqueId** when running the Calendar Server configurator, because the **merge-config** script resets this. For example:

   ```
   davadmin config modify -u admin -o davcore.uriinfo.permanentuniqueid -v davuniqueid
   ```

3. Use the **davadmin** command to both obtain the value of and modify the **davcore.uriinfo.subjectattributes** configuration parameter to add **davUniqueId** to its list of attributes value.

   For example:

   ```
   davadmin config list -u admin -o davcore.uriinfo.subjectattributes
   davcore.uriinfo.subjectattributes: cn davstore icsstatus mail mailalternateaddress
   nsuniqueid owner preferredlanguage uid objectclass ismemberof uniquemember memberurl
   mgrprfc822mailmember kind

   davadmin config modify -u admin -o davcore.uriinfo.subjectattributes -v "cn davstore
   icsstatus mail mailalternateaddress davuniqueid owner preferredlanguage uid
   objectclass ismemberof uniquemember memberurl mgrprfc822mailmember kind"
   ```

Performing the Calendar Server Configuration

To perform the Configuration:

> **Note:**
>
> The following steps are applicable only if you use GlassFish Server.

1. Run the **init-config** script to enter the current deployment configuration values.

   That is, when prompted for the Application Server Configuration Details, use the values for your GlassFish Server 3.1.2 installation. Be sure to enter **davUniqueId** if you chose that as the new unique LDAP attribute.

2. Run the **merge-config** script to merge in any other changes that were done to the configuration after the initial configuration, and that are not controlled by the **init-config** script.

   For example, you might have customized configuration parameters in the **davserver.properties** file.

3. If necessary, resolve any problems.

   If the upgrade detects a problem during the post-upgrade phase with merging configuration files, reconcile the merged files.

davadmin account upgrade Operation

If you have previously upgraded to Calendar Server 7.0.4.14.0, you should have already performed this step. The **davadmin account upgrade** operation sets the next presence triggers for all existing events in the future. You must run **davadmin account upgrade** after upgrading Calendar Server to set the presence triggers for existing future events.

To set presence triggers after upgrading:

```
davadmin account upgrade
```

Currently, **davadmin account upgrade** does not print a message either while it is running or when it completes. The **davadmin account upgrade** command could take some time to complete, so allow it to finish. While **davadmin account upgrade** is running, it does not impact any Calendar Server functionality.

Post-Upgrade Tasks

Complete all of the following post-upgrade tasks after upgrading Calendar Server, if necessary. See "Calendar Server Post-Installation Tasks" for more information.

10
Creating a Calendar Server Coexistent Deployment

This chapter describes how to set up Oracle Communications Calendar Server in an existing Sun Java Calendar Server 6 deployment. In this environment, you have some users who have migrated to Oracle Communications Calendar Server and some users that still exist on Calendar Server 6. In such a deployment, you enable both free/busy lookup and iCalendar Message-Based Interoperability Protocol (iMIP) invitation between the two Calendar Server deployments. That is, users should have the capability to check free/busy information of users on any server, and the capability to invite any user. Invitations to users on a different server would be delivered by using iMIP.

> **Note:**
>
> In this coexistent deployment, users are either on Oracle Communications Calendar Server or Calendar Server 6. They do not have calendars on both systems. Migration from Calendar Server 6 to 8 is not supported from Calendar Server 8.0.0.8.0 onwards.

For more information on performing a Calendar Server migration, see "Migrating to Calendar Server".

Calendar Coexistence Overview

For coexistence to work, each calendar server needs to be able to determine if users are on Calendar Server 6 or Oracle Communications Calendar Server.

On Oracle Communications Calendar Server servers:

- Users are identified as Oracle Communications Calendar Server users by the presence in their Directory Server LDAP entry of a particular attribute which is configured as described in "Configuring the Oracle Communications Calendar Server Environment". For more information on how Calendar Server uses Directory Server, see the topic on Calendar Server and Directory Server integration in *Calendar Server Concepts*.
- All other users that are part of the deployment but do not have this attribute set are considered Calendar Server 6 users.

On Calendar Server 6 servers:

- Users are identified as Calendar Server 6 users if they already have a calendar created in the data base.
- All other users are considered Oracle Communications Calendar Server users.

For coexistence to function correctly, you must disable user auto-provisioning on all Calendar Server 6 servers.

In addition, the same unique ID attribute that you use for Oracle Communications Calendar Server users needs to be present in all Calendar Server 6 users too. This attribute defines the unique value used as the database identifier for each account. This value is used internally to identify a user in other user's access control entries, subscription entries, and so on. The attribute chosen as the unique ID attribute must be present in all user, group, and resource LDAP entries for the deployment.

For example, if Calendar Server 6 uses **uid** as its unique identifier and you configure Oracle Communications Calendar Server to use **davuniqueid** as the unique identifier, you must add the new unique identifier not only to the users migrated to Oracle Communications Calendar Server, but also to the users remaining on Calendar Server 6 for co-existence to work.

For information on migrating a Calendar Server 6 deployment to Oracle Communications Calendar Server, see "Migrating to Calendar Server".

Minimum Software Requirements

- Calendar Server 6.3: At least patch level 44.
- Oracle Communications Calendar Server: At least version 7 Update 1.

Configuring the Calendar Server 6 Environment

Perform the following steps on the Calendar Server 6 host.

1. Patch the server to the following:
   - 121657-37 - Solaris SPARC
   - 121658-37 - Solaris x86
   - 121659-37 - Red Hat Linux
2. Edit the **ics.conf** file as follows:

   For Solaris, edit **/etc/***CalendarServer_home***/SUNWics5/config/ics.conf**.

   For Red Hat Linux, edit **/etc/***CalendarServer_home***/calendar/config/ics.conf**.

   - Set **service.http.caldavcompatible = "yes"**.
   - Set the option for free/busy redirection to point to the Oracle Communications Calendar Server server's free/busy URL. If the Oracle Communications Calendar Server servers have a multiple front-end and back-end setup, use any front-end's information. The **service.wcap.freebusy.redirecturl** parameter must include the port number of the Oracle Communications Calendar Server host.

     For example:

     ```
     service.wcap.freebusy.redirecturl = "http://cs7u1server.example.com:8080/
     davserver/wcap/get_freebusy.wcap"
     ```

   - Disable autoprovisioning by setting the value of **local.autoprovision**, **user.invite.autoprovision**, **group.invite.autoprovision**, and **resource.invite.autoprovision** to **no**.

     For example, the changes resemble the following:

     ```
     service.http.caldavcompatible = "yes"
     service.wcap.freebusy.redirecturl = "http://cs7u1server.example.com:8080/
     davserver/wcap/get_freebusy.wcap"
     local.autoprovision ="no"
     ```

```
user.invite.autoprovision = "no"
group.invite.autoprovision = "no" resource.invite.autoprovision = "no"
```

> **Note:**
>
> In the preceding examples, the **service.wcap.freebusy.redirecturl** is directed to an Oracle Communications Calendar Server host, with a port assignment of 8080.

3. Restart the Calendar Server 6 server:

   ```
   cd /opt/sun/comms/calendar/SUNWics5/cal/sbin
   stop-cal
   start-cal
   ```

4. For migrated Calendar Server 6 users, perform the following steps on the Calendar Server 6 host:

   a. Delete Calendar Server 6 calendars by using **cscal delete -o** *uid*.

   b. Set the LDAP attribute **icsAllowedServiceAccess** to **-http:all**. The setting ensures that Calendar Server 6 users no longer log in by mistake and cause calendars to be autocreated.

Configuring the Oracle Communications Calendar Server Environment

Perform the following steps on the Oracle Communications Calendar Server host.

1. Change to the *CalendarServer_home***/config** directory and either edit or create the **ischeduledomainmap.properties** file.

2. Add a line that contains your domain name and the Calendar Server 6 URL to connect to for Calendar Server 6 user inquiry. If the Calendar Server 6 host is set up for multiple front ends and back ends, you can use any front-end server information.

   For example:

   ```
   example.com=http://cs6server.example.com:8080/ischedule/
   ```

3. Set **davcore.scheduling.localuserattr** parameter, which identifies an LDAP entry as that belonging to an Oracle Communications Calendar Server account. The recommended value is **davstore**.

   For example:

   ```
   davadmin config modify -o davcore.scheduling.localuserattr -v davstore
   ```

4. Add the same value that you used for the **davcore.scheduling.localuserattr** parameter to the **davcore.uriinfo.subjectattributes** parameter.

   For example:

   ```
   davadmin config modify -o davcore.uriinfo.subjectattributes -v davstore
   ```

   If you are using an attribute other than **davstore** for the **davcore.scheduling.localuserattr** parameter, add it to the **davcore.uriinfo.subjectattributes** parameter too. Use the **davadmin config** command to first retrieve the current value of **davcore.uriinfo.subjectattributes**, then add the new attribute at the end of the list.

For example:

```
davadmin config -o davcore.uriinfo.subjectattributes
davcore.uriinfo.subjectattributes: cn davstore icsstatus mail

davadmin config -o modify davcore.uriinfo.subjectattributes -v "cn davstore
icsstatus mail xcs7user"
```

5. On the Directory Server, use an LDAP client to verify that the LDAP attribute defined is present for all users who are on the Oracle Communications Calendar Server server. Also verify that the attribute is not present for users on the Calendar Server 6 host. If using **davStore** and it is a simple single back-end deployment, populate it with the value **defaultbackend**.

Workflow Description of This Configuration

This section contains the following topics:

- About the Calendar Server 6 Configuration
- About the Oracle Communications Calendar Server Configuration

About the Calendar Server 6 Configuration

On a Calendar Server 6 host, setting the **service.http.caldavcompatible** configuration parameter to **yes**, enables the capability to handle iSchedule style free/busy requests from the Oracle Communications Calendar Server host for users still on Calendar Server 6. The value for the **service.wcap.freebusy.redirecturl** configuration parameter specifies the free/busy redirect URL returned by the Calendar Server 6 host for an invitee not found in its local database. Setting the automatic provisioning configuration parameters to **no** (**local.autoprovision**, **user.invite.autoprovision**, **group.invite.autoprovision**, and **resource.invite.autoprovision**) ensures that on a scheduling invitation request, if the invitee calendar is not found on the local database, an iMIP invitation is sent instead of creating an account in the database.

About the Oracle Communications Calendar Server Configuration

When a scheduling invitation or free/busy request comes to an Oracle Communications Calendar Server host, the server performs a directory lookup to determine if each of the invitees is local. If the server option **davcore.scheduling.localuserattr** is set, the server additionally checks if the attribute specified by that option is set for each of the invitees found in the directory. If the attribute is set, the user is considered local to the Oracle Communications Calendar Server host. If the attribute is not set, the user is assumed to be on the Calendar Server 6 host.

If a user is on the Calendar Server 6 host, for scheduling free/busy lookup, the server uses the Calendar Server 6 host information in **ischeduledomainmap.properties** and contacts the Calendar Server 6 host to make an iSchedule free/busy request.

Sample request:

```
>> Request <<

   POST /ischedule/ HTTP/1.1
   Host: cs6server.example.com
   Content-Type: text/calendar
   Content-Length: xxxx
```

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Sun Microsystems/Sun Calendar Server 7.0-0.13//EN
METHOD:REQUEST
BEGIN:VFREEBUSY
DTSTAMP:20040901T200200Z
ORGANIZER:mailto:cs7user@example.com
DTSTART:20040902T000000Z
DTEND:20040903T000000Z
UID:34222-232@example.com
ATTENDEE;CN=Cal7 User:mailto:cs7user@example.sun.com
END:VFREEBUSY
END:VCALENDAR
```

The response is incorporated into the scheduling free/busy response made by the Oracle Communications Calendar Server host. If a user is on a Calendar Server 6 host, for scheduling an invitation, an iMIP invitation is sent by the Oracle Communications Calendar Server host.

11

Migrating to Calendar Server

This chapter describes how to migrate from Sun Java Calendar Server 6 to Oracle Communications Calendar Server. This chapter is applicable only for Calendar Server setup which is performed on GlassFish Server as a container.

> **Note:**
>
> If you are migrating Calendar Server deployment from GlassFish Server to WebLogic Server, perform the following steps:
>
> 1. Ensure to upgrade your existing Calendar Server deployed on GlassFish Server to Calendar Server version 8.x (the recommended version is, 8.0.0.3.0).
> 2. Set up WebLogic Server on a new server with Calendar Server version 8.0.0.4.0 or above which has WebLogic container support.
> 3. Ensure to configure your Calendar Server on WebLogic Server deployment with your required backend server accordingly when migration is completed.

Introduction to Migrating

Because the database formats differ between Calendar Server 6 and Oracle Communications Calendar Server, you must migrate your Calendar Server 6 user data. That is, you cannot directly upgrade from Calendar Server 6 to Oracle Communications Calendar Server. You use the Oracle Communications Calendar Server **davadmin migration** command to migrate the data.

The **davadmin migration** command migrates all relevant Calendar Server 6 data and properties, including calendar data, invitations, ACLs, subscriptions, and so on. There is no migration or updating of LDAP data stored in the Directory Server. Instead, Calendar Server 6 calendar-related properties, such as notifications, previously stored in LDAP, are now stored in the Oracle Communications Calendar Server back-end database, which can be either MySQL Server or Oracle Database. For more information on how Calendar Server uses Directory Server, see the topic on Calendar Server and Directory Server integration in *Calendar Server Concepts*.

> **Note:**
>
> Domain level ACLs are set in LDAP but using a different set of attributes and formats as described in *Calendar Server Security Guide*. The migration does not migrate domain level ACLs.

The migration does not check such items as event invitees, event owners, or delegated owners. The migration transfers whatever information is found in the LDAP directory to the

Oracle Communications Calendar Server database. Additionally, if you migrate the same user multiple times, the user does not end up with duplicate events in the migrated calendar.

The calendar data migration process assumes the following conditions:

- You can allow a reasonable amount of downtime to complete the migration.
- You can perform a trial run to check results before doing the actual migration.
- You can examine and fix events and todos that failed to migrate, by manually updating **ics** files and importing them.

  In general, when fixing and importing make sure the values that used to be **calid** are changed to their **mail** equivalent in all **ics** components.
- Any updates to already scheduled events are not implicit after the migration.
- If your deployment consists of multiple domains, ACIs are configured to enable the Calendar Server administrator ID, set by the **service.siteadmin.userid** configuration parameter (default is **calmaster**), for search and read access to non-default domains.

> **Note:**
>
> You can also have a coexistent deployment of Calendar Server 6 and Oracle Communications Calendar Server hosts. For more information, see "Creating a Calendar Server Coexistent Deployment".

How the Migration Works

The migration utility performs the following sequence of events:

1. Communicates with the Oracle Communications Calendar Server host by using the JMX protocol and invoking the **AdminMigration** process.
2. Creates a log file for that migration task, which is returned to the **davadmin** client.
3. The **AdminMigration** process invokes the Migration Service, which gets the list of users to be migrated from LDAP or passed in from the migration client, and builds a list of users.
4. Starts a migration thread for each user in the list, until it reaches the **serverlimits.maxmigrationthreads** setting. Each migration thread then performs the migration. When done, each thread is reassigned a new user.
5. Each migration thread uses the Migration HTTP client and authenticates to the Calendar Server 6 host by using the provided **administrator** settings and gets a session ID. This session ID is reused for all further WCAP requests to the Calendar Server 6 host, for this particular user.
   a. First, the user's list of calendars and other preferences are fetched by using the **get_userprefs** command.
   b. For each of the user's calendars, the calendar properties are fetched by using the **get_calprops** command.
   c. Then, using the **fetch_by_range** command, events/tasks uids, and type, for the specified period are fetched.

      A map of the uids is made and individual fetches (**fetch_event_by_id** or **fetch_todos_by_id**) are done to get actual data. Fetch is done with the **recurring=1** flag set, to get the master and exceptions as one record. This data is stored in the user's Oracle Communications Calendar Server back-end database, as specified by the LDAP **davStore** attribute. The fetched data is massaged to fit the format for the Oracle Communications Calendar Server host before being stored. Different mapping helper functions perform the data massaging.

6. The user's calendar is automatically created before adding any events or tasks. The fetched calendar properties are mapped and used to update the properties of the newly created calendar.
7. The migration does not overwrite or duplicate any calendar data. When the migration service tries to write data and finds an already existing calendar entry, it just ignores it and continues.

> **Note:**
>
> If a calendar being migrated from Calendar Server 6 already exists on Oracle Communications Calendar Server because it was previously migrated or was explicitly created, the migration resets the calendar's properties to that of the values of the original Calendar Server 6 calendar.

8. Any failed event or task is logged, and the migration task continues.
9. When migration is done, the calendar migration status is set in the migration log file, with more details on what was migrated.
10. The migration steps are repeated for every calendar of the user.
11. When done with one user, the migration thread removes the user from the user list map and picks up the next unmarked user in the list, if any.

Migration Notes

- You migrate all accounts (users and resources) either by providing the **migration** command with the accounts' mail addresses, or by specifying the LDAP base and filter that finds the information on the users and resources in Directory Server. Resource calendars are not migrated by just migrating the resource owner, but by explicitly migrating the resource account itself.
- Resource Owner: The migration sets the resource owner to the value of the primary owner. If the primary owner is not set, the migration uses the LDAP owner field. If there is no LDAP owner, the migration does not set an owner for the resource.
- An account cannot be partially migrated. The process migrates all calendars under that account, as well as subscription lists, calendar settings such as access controls, owner list, notification settings, freebusy settings, and so on.
- Because subscription lists are migrated without checking if the calendars pointed to are migrated as well, some broken subscription links could result until all calendars have been migrated.
- The configuration parameter, **davcore.autocreate.rescalcomponents**, enables migrated resource calendars to allow the **VEVENT** and **VTODO** components that were permitted in Calendar Server 6. (The default value for this parameter is **VEVENT**.) To ensure that Calendar Server 6 resource calendars correctly have all their **VEVENT** and **VTODO** information migrated to Oracle Communications Calendar Server, set this parameter to **VEVENT VTODO** before doing the migration, if your deployment has resource calendars that contain more than just events. For more information, see the topic on Calendar Server configuration parameters in *Calendar Server System Administrator's Guide*.

High-Level Migration Steps

Migrating from Calendar Server 6 to Oracle Communications Calendar Server involves the following high-level steps:

1. Applying the most current Calendar Server 6 patch to fix migration-related issues.
2. Making a list of users to be migrated.
3. Informing users of migration window and down-time.
4. In a multiple domain deployment for LDAP Schema 1 environments, if not already done, configuring LDAP ACIs so that the Calendar Server administrator ID, set by the **service.siteadmin.userid** configuration parameter (default is **calmaster**), has search and read access to non-default domains

   For example:

   a. Ensure that the **icsDomainNames:***non-default domain* attribute is added to the default domain in the **dc** tree.

   b. Ensure that the **icsExtendedDomainPrefs: domainaccess=***cal_svr_admin_id@defaultdomain***ˆaˆrˆg** attribute is added to the non-default domain in the **dc** tree.
5. Setting the **service.http.migrationcompatible** parameter to **yes** (and restarting the Calendar Server).
6. Readying the Oracle Communications Calendar Server deployment for migration by setting the required configuration options.
7. Running the **populate-davuniqueid** script to update users, groups, and resources in LDAP with the unique identifier attribute.
8. Updating the LDAP attribute **davStore** for each user being migrated, to indicate which back-end Oracle Communications Calendar Server host to use.

   > **Note:**
   >
   > Not setting this attribute means the back-end host is local to the Oracle Communications Calendar Server installation.
9. Performing a backup of the user calendars that are being migrated.
10. Creating a file with the list of mail addresses of users to be migrated or determining the LDAP filter to use.
11. Setting the Calendar Server 6 host to "Read Only" mode as described in "To Put a Database in Read-only Mode" in *Sun Java System Calendar Server 6.3 Administration Guide*.
12. Migrating users by running the **davadmin migration** script and with the user list file created or the LDAP base and filter.

    The default action for **davadmin migration** is **migrate** and so can be omitted.
13. Verifying migration status.
14. For successfully migrated users, performing the following steps:

    a. Deleting Calendar Server 6 calendars by running **cscal delete -o** *uid*.

   b. Setting the LDAP attribute **icsAllowedServiceAccess** to **-http:all**.

15. Fixing and importing any failure log files.
16. Notifying migrated users and informing them how to access the Oracle Communications Calendar Server deployment.

Migrating From an SSL-Enabled Calendar Server 6 Deployment

For an Oracle Communications Calendar Server host to communicate with a Calendar Server 6 host over SSL, the Oracle Communications Calendar Server host needs to have the Calendar Server 6 host's certificate available. You do this by exporting the certificate from the Calendar Server 6 host and importing it into the Java runtime environment that is used by the Oracle Communications Calendar Server host.

1. To export a certificate from a Calendar Server 6 host, run the following **certutil** command:

   ```
   certutil -L -a -n alias_name -d CS6_config_directory > CS6_certificate_file.rfc
   ```

   where:

   - *alias_name*: Specifies the alias name of the certificate in the Calendar Server 6 Application Server NSS database
   - *CS6_config_directory*: Specifies the path to the Calendar Server 6 host's **config** directory
   - *CS6_certificate_file.rfc*: Specifies the path to the output file for the certificate

2. Once you have the **.rfc** file, transfer it to the Oracle Communications Calendar Server host and import it into the Java runtime environment that the host is using.

   The Java runtime file that holds the certificates is called **cacerts**. It should have a path similar to **/usr/jdk/latest/jre/lib/security/cacerts**. The import command used to update the cacerts file is: **keytool -import -alias** *alias_name* **-keystore /usr/jdk/latest/jre/lib/security/cacerts -file** *CS6_certificate_file.rfc*

3. Restart Oracle GlassFish Server for these changes to take affect.

Migration Logging

The **davadmin migration** command returns a **log_tag** string, which is the path to the **master_log** file on the server host that contains the current state of that migration. The current state includes a list of migrated users and migration status. This **master_log** file is synchronously updated as the migration progresses.

The **davadmin migration status -Glog_tag** command returns the contents of this **master_log** file and can be used to view the current state of the migration. You can also view the **master_log** file by using UNIX commands such as **cat**, **tail**, **vi**, and so on. The migration is finished when the last line in the **master_log** file is:

```
Migration complete.
```

Here is an example **log_tag** string:

```
/var/opt/sun/comms/davserver/logs/davserver_migration/2010-06-29_153301/master_log
```

To view this file while the migration is in progress, use the following command:

```
tail -f /var/opt/sun/comms/davserver/logs/davserver_migration/2010-06-29_153301/
master_log
```

The root of the directory tree that holds the migration information is **davserver_migration** and defaults to the Calendar Server **log** directory. To store the migration information tree in a different directory, use the **-l** option.

Under the **davserver_migration** directory is a directory named by a time stamp. This directory is created each time that you start a migration operation and the time stamp reflects when you started that migration.

The time stamp-named directory contains the **master_log** file and a directory for each user migrated.

The user's directory contains a directory for each of that user's calendars, including the default calendar and any secondary calendars. The user's directory also contains a **trace_information** directory if the **-c** option was used on the command line. The **trace_information** directory contains files that show the results of commands issued to the Calendar Server 6 host. Use this information in these files to diagnose migration problems.

Each calendar directory contains the **calendar_log** file and **.ics** files. The **calendar_log** contains information about the migration of that particular calendar. The **.ics** files contain the iCalendar data for any event or todo that failed to migrate. If a migration failed, you might be able to fix the iCalendar data in these **.ics** files and import them into the new calendar.

Set the Oracle Communications Calendar Server log level to **FINE** during migration, for easy detection of issues.

Use the **-c** or **--capture** flag to further debug issues that might come up during migration. Migration logging can be quite verbose and takes up lots of disk space.

How the Migration Reformats the Calendar Server 6 Data

To ensure that the Calendar Server data store does not accept non-iCal compliant data, the migration program manipulates or reformats migrated data as follows.

- Email alarms require a description and summary. If either or both are missing, a new value is constructed from the event or task's summary and added.
- Display alarms require a description. If missing, a new value is constructed from the event or task's summary and added.
- Organizer and attendee values are corrected to be valid **mailto:** URIs.
- If an event's **dtend** is not after its **dtstart**, or if they are not of the same type (date only or timed), the bogus **dtend** is discarded and a new one is constructed. If **dtstart** has a date only value, the new **dtend** is one day after the **dtstart**. If the **dtstart** is a timed value, the new **dtend** is one hour after the **dtstart**.
- If a todo has a **dtstart** that is after its due date, or if the value type of both properties do not match, the bogus **dtstart** is discarded. A new **dtstart** with the same value as due is added.
- If a recurring todo has no **dtstart**, a new one is constructed and added. If due exists, the new **dtstart** is the same as due. If no due exists, **dtstart** is set to the current time.
- If a priority field is missing, it is added for todos with attendees.
- If **dtstart** is missing or is bogus, the relative alarm trigger values of todos, relative to due, are set.
- Time strings in events and todos are converted from Zulu to values in their original timezones, and the relevant **vtimezone** information is included.
- A new alarm attendee property is added for explicitly setting SMS alarms that were set by using the **X-S1CS-SMS-EMAIL** property.

- Unending recurring series are truncated to a count specified by **X-NSCP-RECURRENCE-BOUND** in the Calendar Server 6 host, to retain the same recurrence behavior.
- If **dtend** is before or equal to **dtstart**, it is discarded and a new one is created. For all day events, the new dtend is equal to **dtstart** plus 1 day. For timed events, the new **dtend** is equal to **dtstart** plus one 1 hour.

Troubleshooting the Migration

Topics in this section:

- Count Reported by Tools Differs
- No Events Migrated When The Calendar Contains Events
- Exception on creation of userpref Error
- Back-End Database Errors
- LDAP Error
- Read Timed Out Error
- Error Parsing iCal Data
- Owner Property Error
- Other Migration Errors

Count Reported by Tools Differs

The count reported by the Oracle Communications Calendar Server **migration** utility, for example, "Migrated 341 events in: jsmith," is different from the count reported by some Calendar Server 6 tools, such as **csexport** ("number of events=1436"). This difference occurs because most of the Calendar Server 6 tools report number of events in expanded mode (where each instance of a recurring event counts as one), while the **migration** utility reports the number of events in master plus exception mode.

No Events Migrated When The Calendar Contains Events

If your migration completed successfully but did not migrate events or tasks from a calendar that does contain events or tasks, check to see if the **calmaster** ID has permission to access the calendar events and tasks for that user.

For example, if **user1** on Calendar Server 6 resembles the following:

```
cscal -v list user1
user1@example.com: owner=user1@example.com status=enabled
...
number of events=27
number of tasks=0
number of deleted events=0
number of deleted tasks=0
number of deleted recurring events=0
number of deleted recurring tasks=0
```

but when migrated to Oracle Communications Calendar Server produces the following:

```
davadmin migration -u admin -a user1@example.com -X calmaster -L cs6server.example.com:80

[user1@example.com] Creating calendar user1 with name: null
[user1@example.com] Migrated 0 events and 0 tasks in: user1
```

```
[user1@example.com] Subscribed to
[user1@example.com] Migration completed successfully.
```

you should perform the following steps:

1. On the Calendar Server 6 host, look in the **http.log** file log for entries similar to the following. Such entries indicate that the **calmaster** ID is denied access to **user1**'s data:

   ```
   [20/Nov/2011:15:54:11 -0800] cs6server cshttpd[14567]: Account Information: login
   [123.10.10.10] calmaster plaintext sid=DCGF1veS5U8
   [20/Nov/2011:15:54:11 -0800] cs6server cshttpd[14567]: General Error: ac:
   Fetchcomponents: User "calmaster" denied access on fetching from calendar "user1".
   [20/Nov/2011:15:54:11 -0800] cs6server cshttpd[14567]: General Error:
   calstore_fetchcomponents_by_range(): call to ac_fetchcomponents() returning err = 13.
   [20/Nov/2011:15:54:11 -0800] cs6server cshttpd[14567]: General Error:
   calstore_fetchcomponents_by_range(): db error (pError->iErr) = 28.
   ```

2. On the Calendar Server 6 host, check that the **ics.conf** file contains the following two parameters:

   ```
   service.siteadmin.userid
   service.virtualdomain.support
   ```

   a. For non-virtual domain data, the parameters should be set to the following values:

      ```
      service.virtualdomain.support = "no"
      service.siteadmin.userid="calmaster"
      ```

      Here, **calmaster** is an example. Use your site's Calendar Server administrator ID.

   b. For virtual domain data, the parameters should be set to the following values:

      ```
      service.virtualdomain.support = "yes"
      service.siteadmin.userid="calmaster@example.com"
      ```

      Again, **calmaster@example.com** is an example. Use your site's Calendar Server administrator ID.

3. If the two Calendar Server 6 configuration parameter values are not consistent, then permission problems arise and the Calendar Server administrator ID is not allowed access to the calendars that are to be migrated.

To correct the problem:

1. Change the Calendar Server 6 configuration parameters as previously described.
2. Stop and restart calendar services on the Calendar Server 6 host.
3. Rerun the migration.
4. Check the migration output to verify that some number of events were migrated.
5. Check the **http.log** file to ensure that the "denied access" messages no longer occur.

Exception on creation of userpref Error

If your migration produces intermittent errors similar to the following:

```
2010-04-23_155050/master_log:[zw127108@gotmail.example.com ] Exception on creation of
userpref command for zw127108@gotmail.example.com : URI build exception: Invalid query
```

then rerun the migration for the failed user.

Back-End Database Errors

If your migration produces errors similar to the following:

```
[user@host.example.com] Validation exception: backend throws an error
(com.sun.comms.davserver.backends.BackendException: SQL error: Error in allocating a
connection. Cause: In-use connections equal max-pool-size and expired max-wait-time.
Cannot allocate more connections.(INTERNAL_STORE_ERROR)) while doing nodeExists on  dav/
home/user@host.example.com/calendar/dropbox/
00000000000000000000000000000000405ce44e000063da0000003100004aa0/ while storing:
00000000000000000000000000000000405ce44e000063da0000003100004aa0
```

then you might need to do the following:

1. Reduce the value of **davcore.serverlimits.maxmigrationthreads**.

   The default value is 2. Change the value to 2*(number of CPUs).

2. Change the MySQL connection pool size.

   The default is 32. Change this value to 4*(number of threads). To change this value, edit the **/etc/my.cnf** and increase the MySQL **max_connection** setting, for example, **max_connections = 200**.

LDAP Error

If your migration using an LDAP filter produces an error like:

```
LDAP search failure: error result
```

then your filter might be too broad and you might be running into an LDAP search limit exceeded issue. Try narrowing down your filter. For example, if your filter was:

```
davadmin migration -u admin -X calmaster -L cs6.example.com:8080 -B "o=dirbase" -R
"objectclass=icscalendaruser"
```

change it so that it resembles the following:

```
davadmin migration -u admin -X calmaster -L cs6.example.com:8080 -B "o=dirbase" -R
"(&(uid="a*")(objectclass=icscalendaruser))"
```

If you use this approach, you must run multiple migration commands to complete the migration for the entire directory.

Read Timed Out Error

If your migration produces errors similar to the following:

```
2010-04-19_151418/master_log:[user@host.example.com] Exception on creation of http://
host-cs.example.com:80: Read timed out
```

then you might need to increase the **httpconnectiontimeout** configuration options.

To do so, change the **davcore.serverlimits.httpconnecttimeout** and **davcore.serverlimits.httpsockettimeout** options by using the **davadmin config** command.

Error Parsing iCal Data

If your migration produces errors similar to the following:

```
2010-04-20_094910/master_log:user@host.example.com Error parsing ical data for :
00000000000000000000000000000000486c05b900001c6b000001c400000532: failed to parse -
Error at line 48:Illegal character in opaque part at index 27:
user1@varrius.com\,user@example.com\,user1_smith@varrius.com
```

then you must fix the line giving the error in the **ics** file located in the migration directory and import it.

You can use either the **davadmin import** or **client import** commands to import the file.

The following examples provide other potential errors of this type.

Error Parsing iCal Data Example 1

```
For
Error at line 32:I
illegal character in scheme name at index 6: mailto\:mattp@example.com
Change value to mailto:mattp@example.com
```

Error Parsing iCal Data Example 2

```
For
Error at line 38:I
llegal character in scheme name at index 6: mailto\:mailto\\\\\\\:mailto\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\:mailto\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\ \\\\\\\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\:ms104926@iplanet.com
Change value to mailto:ms104926@iplanet.com
```

Error Parsing iCal Data Example 3

```
For
Error at line 38:Illegal character in opaque part at index 16: mailto:Marco@example
Microsystems
Change value to a vaild email address like mailto:Marco@example.com
```

Owner Property Error

If the migration log shows the message "FAILED TO SET OWNER PROPERTY" when migrating a resource, there might be a problem with the "other owners" of the resource that you are migrating. A known issue with the Calendar Server 6 **csresource create** command could have caused the "other owners" field to be created incorrectly.

If you did the migration using the **-c** option, the log directory contains a **trace_information** directory for the resource. In the **trace_information** directory, search the **getcalprops*** file for the **X-NSCP-CALPROPS-OWNERS** line. If this line contains a comma separated list of users, these "other users" were set up incorrectly in Calendar Server 6. There should be one **X-NSCP-CALPROPS-OWNERS** line for each of the "other owners" for the resource.

If this is the case, correct the problem as follows:

1. Change the "other owners" by using the Calendar Server 6 **cscal** command with the **-y** option and space delimited user names.

   For example:

   ```
   cscal modify -y "" uid
   cscal modify -y "1st owner 2nd owner ..."
   ```

   The first command clears out the "other owners" field and the second sets it correctly.

2. On Oracle Communications Calendar Server, delete the account that you migrated and rerun the migration.

Other Migration Errors

The section describes possible migration errors that you must individually fix and import. The failed **ics** files are located under the **calendar** subdirectory for individual users. Fix then import these files either by using the **davadmin import** command or by having the end users import their own files. In general, when fixing and importing make sure the values that used to be **calid** are changed to their **mail** equivalent in all **ics** components.

Topics in this section:

- Multiple Components with the Same uid but Different Date Type
- Failed to store component: F0095280-BC13-11D9-81F6-000A958EAC78
- Error parsing ical data for: 3d6cd576000000a70000000a00000a55: failed to parse
- Failed to store component: 9DE4F2DA-D020-11D8-9530-000A958A3252
- Failed to store component
- Failed to store component: failed to parse - Error at line 84
- Error parsing ical data for: F0076B82-BC13-11D9-81F6-000A958EAC78

Multiple Components with the Same uid but Different Date Type

```
DATETIME_WITH_TZ,DATETIME_UTC
```

This error happens if two separate events or tasks end up with the same uid and the **dtstart**/ **dtend** types do not match. Separate out the events into two separate **ics** files and import.

For example, this event:

```
BEGIN:VCALENDAR^M
PRODID:-//Sun/Calendar Server//EN^M
METHOD:PUBLISH^M
VERSION:2.0^M
X-NSCP-CALPROPS-LAST-MODIFIED:20031101T061300Z^M
X-NSCP-CALPROPS-CREATED:20010418T022506Z^M
X-NSCP-CALPROPS-READ:999^M
X-NSCP-CALPROPS-WRITE:999^M
X-NSCP-CALPROPS-RELATIVE-CALID:user@example.com^M
X-NSCP-CALPROPS-NAME:Business^M
X-NSCP-CALPROPS-PRIMARY-OWNER:user@example.com^M
X-NSCP-CALPROPS-TZID:America/New_York^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^c^wdeic^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^a^rsf^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^a^^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^c^^g^M
X-NSCP-CALPROPS-RESOURCE:0^M
X-S1CS-CALPROPS-ALLOW-DOUBLEBOOKING:1^M
X-NSCP-WCAP-ERRNO:0^M
BEGIN:VEVENT^M
UID:3bcdec24000054d40000000d00000db5^M
DTSTAMP:20100503T225631Z^M
SUMMARY: Meeting^M
CREATED:20011029T161741Z^M
LAST-MODIFIED:20091009T200936Z^M
PRIORITY:0^M
SEQUENCE:0^M
```

```
CLASS:PUBLIC^M
STATUS:CONFIRMED^M
TRANSP:OPAQUE^M
RRULE:FREQ=MONTHLY;WKST=MO;COUNT=41;INTERVAL=1;BYDAY=1FR^M
DTSTART;TZID=America/New_York:20011102T083000^M
DTEND;TZID=America/New_York:20011102T123000^M
X-S1CS-RECURRENCE-RDATELIST:20020405T133000Z^M
.....
END:VEVENT^M
BEGIN:VEVENT^M
UID:3bcdec24000054d40000000d00000db5^M
RECURRENCE-ID:20020405T133000Z^M
DTSTAMP:20100503T225631Z^M
SUMMARY:A Bday^M
DTSTART;VALUE=DATE:20020410^M
DTEND;VALUE=DATE:20020411^M
CREATED:20011017T203756Z^M
LAST-MODIFIED:20091009T200335Z^M
PRIORITY:0^M
SEQUENCE:0^M
CLASS:PUBLIC^M
STATUS:CONFIRMED^M
TRANSP:TRANSPARENT^M
X-NSCP-ORIGINAL-DTSTART:20020405T133000Z^M
X-NSCP-LANGUAGE:en^M
X-NSCP-DTSTART-TZID:America/New_York^M
X-NSCP-TOMBSTONE:0^M
X-NSCP-ONGOING:0^M
X-NSCP-GSE-COMPONENT-STATE;X-NSCP-GSE-COMMENT=REQUEST-COMPLETED:131074^M
END:VEVENT^M
....

END:VCALENDAR
```

can be split to the first master event with its timed exceptions and a new master event for the all day event:

```
BEGIN:VCALENDAR^M
PRODID:-//Sun/Calendar Server//EN^M
METHOD:PUBLISH^M
VERSION:2.0^M
X-NSCP-CALPROPS-LAST-MODIFIED:20031101T061300Z^M
X-NSCP-CALPROPS-CREATED:20010418T022506Z^M
X-NSCP-CALPROPS-READ:999^M
X-NSCP-CALPROPS-WRITE:999^M
X-NSCP-CALPROPS-RELATIVE-CALID:user@example.com^M
X-NSCP-CALPROPS-NAME:Business^M
X-NSCP-CALPROPS-PRIMARY-OWNER:user@example.com^M
X-NSCP-CALPROPS-TZID:America/New_York^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^c^wdeic^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^a^rsf^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^a^^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^c^^g^M
X-NSCP-CALPROPS-RESOURCE:0^M
X-S1CS-CALPROPS-ALLOW-DOUBLEBOOKING:1^M
X-NSCP-WCAP-ERRNO:0^M
BEGIN:VEVENT^M
UID:3bcdec24000054d40000000d00000db5^M
RRULE:FREQ=YEARLY
DTSTAMP:20100503T225631Z^M
SUMMARY:Camille Bday^M
DTSTART;VALUE=DATE:20020410^M
```

```
DTEND;VALUE=DATE:20020411^M
CREATED:20011017T203756Z^M
LAST-MODIFIED:20091009T200335Z^M
PRIORITY:0^M
SEQUENCE:0^M
CLASS:PUBLIC^M
TRANSP:TRANSPARENT^M
X-NSCP-LANGUAGE:en^M
X-NSCP-TOMBSTONE:0^M
X-NSCP-ONGOING:0^M
X-NSCP-GSE-COMPONENT-STATE;X-NSCP-GSE-COMMENT=REQUEST-COMPLETED:131074^M
END:VEVENT^M
END:VCALENDAR
```

Failed to store component: F0095280-BC13-11D9-81F6-000A958EAC78

```
validation error: PERCENT-COMPLETE with invalid value: -1
```

Edit **PERCENT-COMPLETE** to be between 0 and 100.

Error parsing ical data for: 3d6cd576000000a70000000a00000a55: failed to parse

```
Error at line 52: Illegal character in opaque part of index 27:
MAILTO: user.one@example.com/n
```

Remove the trailing {{n} character or any other character that would be illegal in an email address.

Failed to store component: 9DE4F2DA-D020-11D8-9530-000A958A3252

```
validation error: Calendar must contain at least one component
```

In this case, the **ics** file resembles the following:

```
BEGIN:VCALENDAR^M
PRODID:-//Sun/Calendar Server//EN^M
METHOD:PUBLISH^M
VERSION:2.0^M
X-NSCP-CALPROPS-LAST-MODIFIED:20031101T061545Z^M
X-NSCP-CALPROPS-CREATED:20030916T095049Z^M
X-NSCP-CALPROPS-READ:999^M
X-NSCP-CALPROPS-WRITE:999^M
X-NSCP-CALPROPS-RELATIVE-CALID:user@example.com^M
X-NSCP-CALPROPS-NAME:User One^M
X-NSCP-CALPROPS-LANGUAGE:en^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^a^r^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@@o^c^wdeic^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^a^sf^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^c^\^g^M
X-NSCP-CALPROPS-ACCESS-CONTROL-ENTRY:@^p^r^g^M
X-NSCP-CALPROPS-RESOURCE:0^M
X-S1CS-CALPROPS-ALLOW-DOUBLEBOOKING:1^M
X-NSCP-WCAP-ERRNO:59^M
END:VCALENDAR^M
```

You may ignore the file as the event does not exist any longer.

Failed to store component

```
00000000000000000000000000000000472defae0000696e000001460000129f: validation error:
master non recurring
```

Construct an **RRULE** property and add it to the master component.

Failed to store component: failed to parse - Error at line 84

```
Illegal character in scheme name at index:
=BASE64;VALUE=BINARY;X-SICS-FILESIZE=49665:ATTACHFMTTYP//MESSAGE/
OLEXS1CSATTACHIAAA007110202XS1CSCLIENTIAAAec798b438d5469da5af862534819cf2XS1CSFILENAME/
ENCODEIN//BASE64VALU//BINARYXS1CSFILESIZ/40AAAA
```

This error occurs when the attachment is missing. Edit the file and remove the line with the **ATTACH** property.

Error parsing ical data for: F0076B82-BC13-11D9-81F6-000A958EAC78

```
failed to parse - Error at line 52: [EXDATE] Unparseable date: "20000220"
```

This should not happen as long as the Calendar Server 6 host is patched with the most current Calendar Server patch. However, it you do see this error, edit the **EXDATE** property line and add **VALUE=DATE EXDATE;VALUE=DATE:20000220**.

12

Uninstalling Calendar Server

This chapter describes how to uninstall Oracle Communications Calendar Server.

Uninstalling Calendar Server

The **commpkg uninstall** command enables you to uninstall Communications Suite products, such as Calendar Server, as well as shared components. However, the **commpkg uninstall** command does not remove OS patches or shared components installed by **commpkg install**.

To uninstall Calendar Server:

1. Log in as **root**.
2. Change to the *InstallRoot***/CommsInstaller/bin/** directory.
3. Run the **commpkg uninstall** command.
4. Choose Calendar Server from the list of installed Communications Suite components.
5. When prompted, enter **Yes** to continue.

13

Installing Patches

This chapter describes how to install patches on Oracle Communications Calendar Server.

See the patch ReadMe file, included in the patch download, for information about the contents of a patch.

About Patching Calendar Server

Calendar Server patches are posted on the My Oracle Support web site:

`https://support.oracle.com`

> **Caution:**
>
> Always read the patch ReadMe file in its entirety before installing a patch.

Some patches contain fixes and functionality that may not be of any interest to you or may apply to features that you have not installed or purchased. Read the patch ReadMe file to determine if you must install the patch.

Some patches are password protected. To request the password to download a protected patch, open a Service Request on the My Oracle Support web site.

Planning Your Patch Installation

Before installing a patch, verify your version of Calendar Server and ensure the patch is not already installed.Oracle recommends scheduling your patch installation during non-peak hours to minimize the disruption to your operations.Oracle recommends installing a patch on a test system with a copy of your production data before installing the patch on your production system. Test the patch by logging into Calendar Server and verifying the version number of installed components.

Installing a Patch

Oracle Solaris 11 introduced the Image Packaging System (IPS) for software installs and updates. IPS changes the way Unified Communications Suite delivers patches, because IPS does not support the **patchadd** command. On Solaris 11 systems, you must use Automated Release Update (ARU) patches. These patches differ from the older SRV4 Sun-style patches, which are not supported on Solaris 11. You can use ARU patches on other Solaris releases as well. To install a Unified Communications Suite ARU patch, you use the **commpkg upgrade** command.

Installing an ARU Patch

To install an ARU patch on Calendar Server:

1. Back up your Calendar Server back-end database.

   For example, you can use the **davadmin db backup** command.

2. (Optional) To prevent users from connecting during the patch application, you can disable the application server from servicing incoming requests.

   If you use GlassFish Server, see "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

   If you use WebLogic Server, stop the Managed Server or target server on which the application is deployed.

3. Apply the patch by running the following command.

   ```
   commpkg upgrade
   ```

4. Run the Calendar Server **init-config** script to enter the current deployment configuration values.

5. Run the Calendar Server **merge-config** script to merge in any other changes that were done to the configuration after the initial configuration, and that are not controlled by the **init-config** script.

   For example, you might have customized configuration parameters in the **davserver.properties** file.

6. (Optional) Resolve any problems.

   a. If the patch installation detects a problem during the pre-patch phase, correct the problem and re-apply the patch.

   b. If the patch installation detects a problem during the post-patch phase with merging configuration files, reconcile the merged files.

7. Restart GlassFish Server.

Installing an SRV4 Patch

To install an SRV4-style patch on Calendar Server:

1. Back up your Calendar Server back-end database.

   For example, you can use the **davadmin db backup** command.

2. (Optional) To prevent users from connecting during the patch application, you can disable the application server from servicing incoming requests.

   If you use GlassFish Server, see "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

   If you use WebLogic Server, stop the Managed Server or target server on which the application is deployed.

3. Apply the patch by running the **pkgadd** command.

   See the **pkgadd** man page for more information.

4. Run the Calendar Server **init-config** script to enter the current deployment configuration values.

5. Run the Calendar Server **merge-config** script to merge in any other changes that were done to the configuration after the initial configuration, and that are not controlled by the **init-config** script.

   For example, you might have customized configuration parameters in the **davserver.properties** file.

6. (Optional) Resolve any problems.
   a. If the patch installation detects a problem during the pre-patch phase, correct the problem and re-apply the patch.
   b. If the patch installation detects a problem during the post-patch phase with merging configuration files, reconcile the merged files.
7. Restart GlassFish Server.

Installing a Linux Patch

To install a Linux patch on Calendar Server:

1. Back up your Calendar Server back-end database.

   For example, you can use the **davadmin db backup** command.
2. (Optional) To prevent users from connecting during the patch application, you can disable the application server from servicing incoming requests.

   If you use GlassFish Server, see "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

   If you use WebLogic Server, stop the Managed Server or target server on which the application is deployed.
3. Apply the patch by running the **rpm -F** *rpmname* command.

   See the **rpm** man page for more information.
4. Run the Calendar Server **init-config** script to enter the current deployment configuration values.
5. Run the Calendar Server **merge-config** script to merge in any other changes that were done to the configuration after the initial configuration, and that are not controlled by the **init-config** script.

   For example, you might have customized configuration parameters in the **davserver.properties** file.
6. (Optional) Resolve any problems.
   a. If the patch installation detects a problem during the pre-patch phase, correct the problem and re-apply the patch.
   b. If the patch installation detects a problem during the post-patch phase with merging configuration files, reconcile the merged files.
7. Restart GlassFish Server.

Disabling GlassFish Server Incoming Connections During Patch Application

This section describes how to disable GlassFish Server so that it does not respond to incoming requests for connections while patching Calendar Server.

Disabling GlassFish Server Incoming Connections Overview

GlassFish Server needs to be up and running when you apply a patch to Calendar Server. If desired, you can disable GlassFish Server from servicing incoming user connections while patching, so that no data loss occurs during the patch application. You do so by disabling the http listeners, **http-listener-1** and **http-listener-2**, on the running GlassFish instance. In this situation, GlassFish Server is still running, so that the Calendar Server configurator functions, but at the same time it is disallowing incoming connections.

Disabling GlassFish Server Incoming Connections

To disable GlassFish Server from servicing incoming connections:

1. List what the http listener is set to.

```
GlassFish_home/bin/asadmin --host=hostname --port=portnumber --secure=true --
user=admin get server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled
server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled=true
Command get executed successfully.
```

2. Disable the http listener.

```
GlassFish_home/bin/asadmin --host=hostname --port=portnumber --secure=true --
user=admin set server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled=false
server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled=false
Command set executed successfully.
```

Re-enabling GlassFish Server http Listeners

After you have completed applying the Calendar Server patch, and have finished running additional configuration steps, re-enable the http listeners.

```
GlassFish_home/bin/asadmin --host=hostname --port=portnumber --secure=true --user=admin
set server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled=true
server-config.network-config.network-listeners.network-listener.http-
listener-1.enabled=true
Command set executed successfully.
```

A

commpkg Reference

This appendix provides information about the Oracle Communications Calendar Server **commpkg** command.

Overview of the commpkg Command

The **commpkg**, command, also referred to as the installer, comprises several commands (verbs) that enable you to install, uninstall, and upgrade Calendar Server software and its shared components. The **commpkg** command is installed in the directory in which you extract the product software.

Syntax

```
commpkg [general_options] verb [verb_options]
```

Table A-1 describes the **commpkg** command general options.

Table A-1 commpkg Command General Options

Option	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--OSversionOverride	Overrides the operating-system version check.
--fixEntsys [**y** | **n**]	Fixes an invalid Sun Java Enterprise System (Java ES) **entsys symlink**, making the link point to the latest Java version upgraded by **commpkg**. The Java ES symlink is located in **/usr/jdk/entsys-j2se**. Choose **--fixEntsys y** to fix the Java ES symlink to the Java files. If you do not specify this switch, **commpkg** prompts you if the symlink is invalid. However, in silent mode, the default is not to fix the symlink (the equivalent of using a value of n). To fix the symlink in silent mode, type **commpkg install --fixEntsys y --silent** *INPUTFILE* on the command-line.

Table A-2 describes the **commpkg** command verbs.

Table A-2 commpkg Command Verbs

Verb	Description
install	Performs software installation.
uninstall	Uninstalls software but does not remove OS patches or shared components installed by **commpkg install**.
info	Displays product information on the paths (also known as *installroots*) where Calendar Server is installed, and the products that are installed in those paths.
upgrade	Performs software upgrade.

Table A-2 (Cont.) commpkg Command Verbs

Verb	Description
verify	Verifies installed product.

install Verb Syntax

```
commpkg install [install_options] [ALTROOT|name]
```

> **Tip:**
>
> Installing Only Shared Components: To install just the product's shared components, launch the installer then prefix your product selection with a tilde (~). You can type multiple selections by using a comma to separate the entries.

Table A-3 describes the **commpkg install** verb options.

Table A-3 commpkg install Options

commpkg install Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Does not apply operating system patches during product installation.
--excludeSC	Does not install, upgrade, or patch any shared components.
ALTROOT | *name*	Use this option to install multiple instances of the product on the same host or Oracle Solaris zone. You can also use this option to perform a side-by-side upgrade of the product. This option is available on Solaris only. Specifies an alternate root directory for a multi-instance installation. The *InstallRoot* (the top-level installation directory for all products and shared components) is the alternate root. If you specify a *name*, it will be a friendly name associated with the *ALTROOT* that is registered in the software list. If you specify the *name* and it exists in the software list, the corresponding *ALTROOT* is used. If you also specify the **--installroot** option, it must correspond to the entry in the software list. If you specify name and it does not exist in the software list, it is added to the software list. Specifying any *name* other than “” implies an **ALTROOT**. A value for *name* of “” is reserved for the default root.
--installroot	Specify location of INSTALLROOT, the top level installation directory for all products and shared components. The top-level installation directory for individual products are subdirectories under INSTALLROOT. Default is **/opt/sun/comms**.
--distro *path*	Specifies the *path* to packages or patches for the products. Default: Location of **commpkg** script

Table A-3 (Cont.) commpkg install Options

commpkg install Options	Description
--silent *INPUTFILE*	Runs a silent installation, taking the inputs from the *INPUTFILE* and the command-line arguments. The command-line arguments override entries in the *INPUTFILE*. Installation proceeds without interactive prompts. Use **--dry-run** to test a silent installation without actually installing the software. Specify **NONE** for *INPUTFILE* to run in silent mode without using an input file. When you specify **NONE**, the installation uses default values.
--dry-run or **-n**	Does not install software. Performs checks.
--upgradeSC [**y**|**n**]	Upgrades or does not upgrade shared components as required. If this option is not specified, you are prompted for each shared component that must be upgraded by using package removal and installation. Default: **n** **Caution:** Upgrading shared components by using package removal and installation is irreversible. However, if you do not upgrade required shared components, products might not work as designed. The **--excludeSC** flag has precedence over this flag.
--auditDistro	Audits the installation distribution to verify that the patches and packages matches the versions in the product files internal to the installer.
--pkgOverwrite	Overwrites the existing installation package. You might use this option when you are installing a shared component in a global zone where either the shared component does not exist in a global zone, or the shared component exists in the whole root zone. The default is not to override the existing package. In general, shared components should be managed in the global zone.
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.
--skipOSLevelCheck	(Solaris only) The installer always checks that the operating system is at a certain minimum patch level. Using this option skips the check.

uninstall Verb Syntax

```
commpkg uninstall [verb_options] [ALTROOT|name]
```

Table A-4 describes the **commpkg uninstall** verb options.

> **Note:**
>
> A fast way to uninstall a product that was installed in an alternate root is to simply remove the entire alternate root directory.

Table A-4 commpkg uninstall Options

commpkg install Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--silent *INPUTFILE*	Runs a silent uninstall, taking the inputs from the *INPUTFILE* and the command-line arguments. The command-line arguments override entries in the *INPUTFILE*. Uninstall proceeds without interactive prompts. Use **--dry-run** to test a silent installation without actually installing the software.
--dry-run or **-n**	Does not install software. Performs checks.
ALTROOT | *name*	Use this option to uninstall multiple instances of the product on the same host or Oracle Solaris zone. You can also use this option to perform a side-by-side upgrade of the product. This option is available on Solaris only. Specifies an alternate root directory for a multi-instance uninstallation. The *InstallRoot* (the top-level installation directory for all products and shared components) is the alternate root. If you specify a *name*, it will be a friendly name associated with the *ALTROOT* that is registered in the software list. If you specify the *name* and it exists in the software list, the corresponding *ALTROOT* is used. If you also specify the **--installroot** option, it must correspond to the entry in the software list. If you specify name and it does not exist in the software list, it is added to the software list. Specifying any *name* other than “” implies an **ALTROOT**. A value for *name* of “” is reserved for the default root.

upgrade Verb Syntax

```
commpkg upgrade [verb_options] [ALTROOT|name]
```

Table A-5 describes the **commpkg upgrade** verb options.

Table A-5 commpkg upgrade Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Does not apply operating system patches during product upgrade.
--excludeSC	Does not install, upgrade, or patch any shared components.
ALTROOT | *name*	This option is available on Solaris only. Specifies an alternate root directory during a multiple host installation. The *InstallRoot* (the top-level installation directory for all products and shared components) is the alternate root. If you specify a *name*, it is an “alias" associated with the alternate root that is registered in the software list. You can use this option to upgrade multiple product instances on the same host or Solaris zone. Additionally, you can use this option to perform a side-by-side product upgrade.

Table A-5 (Cont.) commpkg upgrade Options

Options	Description
--distro *path*	Specifies the *path* to packages and patches for the products.Default path: Location of the **commpkg** command.
--silent *INPUTFILE*	Runs a silent upgrade, taking the inputs from the *INPUTFILE* and the command-line arguments. The command-line arguments override entries in the *INPUTFILE*. Upgrade proceeds without interactive prompts. Use **--dry-run** to test a silent upgrade without actually installing the software. Specify **NONE** for *INPUTFILE* to run in silent mode without using an input file. When you specify **NONE**, the upgrade uses default values.
--dry-run or **-n**	Does not upgrade software but performs checks. This option creates a silent upgrade file in the **/tmp** directory.
--upgradeSC [**y**|**n**]	Indicates whether to upgrade shared components as required. If this option is not specified, you are prompted for each shared component that must be upgraded by the package uninstall/install. Default: **n** **Caution:** Upgrading shared components is irreversible. However, if you do not upgrade required shared components, products might not work as designed. The --**excludeSC** flag has precedence over this flag.
--pkgOverwrite	This option is only for Solaris systems. Overwrites the existing installation package. You might use this option when you are installing a shared component in a global zone where either the shared component does not exist in a global zone, or the shared component exists in the whole root zone. The default is not to override the existing package. In general, shared components should be managed in the global zone.
--components *comp1 comp2...*	Specifies products to be upgraded. Separate each component product with a space. (Thus, the list is a space-delimited set.) To specify each component product, use the mnemonic associated with that product. To display a list of the mnemonics for all the component products, use the **commpkg info --listpackages** command.
--usePkgUpgrade	If the upgrade can be performed by using a patch or an in-place package upgrade, this option uses the in-place package upgrade. The default is to use a patch to upgrade, if possible. **Note:** Patches are used only on Solaris.

verify Verb Syntax

```
commpkg verify [verb_options] [ALTROOT|name]
```

Tip:

When verifying the package installation in an alternate root, be aware that Calendar Server uses the operating system components installed in the default root. Some products might also use shared components in the default root. Thus, verify the package installation in the default root as well.

Table A-6 describes the **commpkg verify** verb options:

Table A-6 commpkg verify Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Do not verify operating system components.
--excludeSC	Do not verify shared components.
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.
ALTROOT | *name*	Use this option to verify multiple instances of the product on the same host or Solaris zone. This option is available on Solaris only. Specify *ALTROOT* or *name* for an alternate root directory on which to perform the package verification.

info Verb Syntax

```
commpkg info [verb_options] [ALTROOT|name]
```

Table A-7 describes the **commpkg info** verb options.

Table A-7 commpkg info Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--clean	Removes entries in the software list. If *ALTROOT* | *name* is specified, the option removes the entry from the software list. If no *ALTROOT* | *name* is specified, the option removes all entries from the software list.
--listPackages	Lists the packages that comprise Calendar Server, shared components, and operating system auxiliary products. This option also displays the mnemonic for Calendar Server and components such as **comm_dssetup.pl**.
--verbose	Prints product information installed in the *installroots*. To print information for a specific *installroot*, run the following command: **commpkg info --verbose** *installroot*
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.

About the Alternate Root

You can install multiple copies of the same product version on the same Solaris machine or Solaris zone by using the alternate root feature of the **commpkg install** command. For example, you might deploy a host with an installation in the default root directory, **/opt/sun/comms**, and a second, separate installation in the **/opt/sun/comms2** alternate root directory. The alternate root installation directory is the top-level directory underneath which the Calendar Server component product and shared components are installed in their respective directories.

Some possible uses for multiple installations include:

1. Performing a side-by-side upgrade.
2. Enabling an installation to be easily moved from one machine to another.

> **Note:**
>
> The alternate root feature is available only on Solaris. This feature is a “power user" feature. If you are interested in installing more than one instance of the same version of Calendar Server on the same physical host, another option is to use Solaris zones. For more information, see "Installing on Solaris OS Zones: Best Practices".

ALTROOT | name Syntax and Examples

You can use the optional *ALTROOT* **|** *name* option with the **commpkg install**, **commpkg upgrade**, **commpkg uninstall**, and **commpkg verify** commands. You use either *ALTROOT* or *name*. The *name* acts as an alias for the alternate root installation path. The *name* is registered in an internal software list maintained by the installer. You can use *name* for the alternate root's path in commands that accept the alternate root. The distinction between the alternate root and name is that the alternate root always begins with a slash (/) whereas the name does not.

Syntax:

```
commpkg [install|upgrade|unistall|verify] [ALTROOT|name]
```

Example 1:

```
commpkg install /opt/sun/comms2
```

In this example, the path **/opt/sun/comms2** is the alternate root, which becomes the top-level directory underneath which Calendar Server software and shared components are installed.

Example 2:

```
commpkg install Comms2
```

In this example, **Comms2** is the name for the alternate root. During the installation process, the installer prompts you to type in the alternate root installation directory.

Example 3:

In this example, you avoid installing the shared components in the alternate root by using the **--excludeSC** option:

```
commpkg install --excludeSC /opt/sun/comms2
```

Example 4:

To install only the shared components, run the **commpkg install** command and select the product you want to install, but prepend a tilde (~).

For example, if you plan to install Calendar Server in the alternate root, you select ~1 during the default installation. This tells the installer to install the dependencies but not the product itself.

Notes on the *ALTROOT* **|** *name* command-line argument:

- Specifying a slash (/) as an alternate root is the same as specifying the default root installation directory. That is, specifying a slash is interpreted by the installer as having specified no alternate root.
- Likewise, specifying “" as an alternate root is interpreted as having specified no alternate root. (The “friendly name" for the default alternate root is “".)
- If you specify the **--installroot** option in addition to *ALTROOT* **|** *name*, the two must match.
- Operating system patches are always installed into the default root (/). Some shared components are installed into the *ALTROOT* and some are installed into the default root (/).
- You can quickly uninstall an *ALTROOT* installation by using the **rm -r** command on the alternate root directory. The next time that you run the **commpkg info** command, the internal software list that maintains the alternate root information is updated.
- You can create as many alternate roots as you like. Running the **commpkg info** command reports on the various alternate roots.

Understanding the Difference Between ALTROOT and INSTALLROOT

The following concepts define an alternate root (ALTROOT):

- An alternate root directory is a Solaris feature that is used by the **commpkg** command to enable multiple product installations on the same host.
- The default alternate root is the standard root directory (/) and is always present.

The following concepts define an installation root (*InstallRoot*):

- An *InstallRoot* is the top-level umbrella installation path for Calendar Server.
- On the default alternate root (that is, /), you can specify an *InstallRoot*.
- On an alternate root, the *InstallRoot* is the same as the alternate root.

Default Root

If you use the default root, the default *InstallRoot* is:

/opt/sun/comms/

Using Both Default Root and Alternate Root

Suppose you want to install one instance of Calendar Server in the **/opt/sun/mycompany/comms/** directory, and another instance of the same product in the **/opt/sun/mycompany/comms2/** directory. You would use the following commands:

For the default root:

```
commpkg install --installroot /opt/sun/mycompany/comms
```

For the alternate root:

```
commpkg install /opt/sun/mycompany/comms2/
```

Running Multiple Installations of the Same Product on One Host: Conflicting Ports

By default, after you initially configure the product on alternate roots, the ports used by the different product installations are the same and thus conflict with each other.

This is not a problem if you install multiple installations of the same product on the same host but only intend to have one instance running at one time. For example, you may perform a side-by-side upgrade scenario in which you plan to stop the old instance before you start the new instance.

However, you may plan to test the new instance while the old instance is still running (and supporting end users). In this scenario, the ports are used simultaneously, and you must take steps to resolve the port conflicts.

B

Calendar Server Configuration Scripts

This appendix provides information about the Oracle Communications Calendar Server configuration scripts.

init-config Script

The **init-config** script enables you to perform an initial configuration of your Calendar Server deployment. Table B-1 describes the **init-config** options.

Table B-1 init-config Options

Option	Description
-f *statefile_name*	Uses the *statefile_name* for setting input values. Use the **DavserverCfgDefaults.properties** file as an example.
-i *hostname*	Uses *hostname* as the FQDN of the current host.
-l	Uses the name returned by the **/bin/hostname** command for name of host. **/bin/hostname** must return an FQDN.
-s	Performs a silent initial configuration, requires the **-f** option.
-S *statefile_name*	Saves state of **init-config** script input to a named *statefile_name*.
-v	Enables verbose mode.
-W	Use this option for WebLogic Server. It is applicable for the initial configuration. **Note**: If you do not use this option, GlassFish Server is used as the default application server.

extractSSLArgs.sh Script

When you configure Calendar Server for the first time with Oracle WebLogic Server in a secure mode, run the **extractSSLArgs.sh** script. This script validates the SSL configuration details in Oracle WebLogic Server and stores the valid details in a format that is required by Calendar Server for all future deployments and processing.

Options

The following table lists the options for validating and storing Oracle WebLogic Server SSL details.

Table B-2 WebLogic Server Options

Option	Description
-u	WebLogic Server Administrator User ID
-p	WebLogic Server Administrator User Password

Table B-2 (Cont.) WebLogic Server Options

Option	Description
-l	WebLogic Server Administrator URL *t3|t3s://FQDN:Weblogic AdminServer (SSL|NonSSL)Port* For example, **t3://hostname.com:7001** Or **t3s://hostname.com:7002**
-r	Runtime directory under which **/config/davadmin.properties** file resides. For example, **/var/opt/sun/comms/davserver/config/davadmin.properties**. You can use this option only on the existing Calendar Server deployment which is already running on WebLogic Server. Use this option if WebLogic Server's keystore settings are modified after Calendar Server is deployed and those settings have to be updated on Calendar Server Configuration.

Running the Script for a Fresh Installation

- Validate the SSL and Keystore configurations on WebLogic Server setup
- Perform the steps in the "Installing and Configuring WebLogic Server" section to set up WebLogic Server in a secure mode.
- Keep the configuration details of WebLogic Server handy.

To validate and store Oracle WebLogic Server SSL details for Calendar Server in a secure mode:

1. Open a new terminal and prepare the terminal by sourcing the **setDomainEnv.sh** script of the Oracle WebLogic Server domain:

   ```
   cd Weblogic_Domain/bin

   source ./setDomainEnv.sh   OR  . ./setDomainEnv.sh
   ```

2. Set the **WLST_PROPERTIES** environment variable depending on the selected Oracle WebLogic Server keystore configuration.

   - If the **CustomIdentityandCustomTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_PROPERTIES="-Dweblogic.security.TrustKeyStore=CustomTrust , -
     Dweblogic.security.CustomTrustKeyStoreFileName=Weblogic_Domain/mytrust.jks"
     ```

     where *Weblogic_Domain/mytrust.jks* is the location of truststore file location.

   - If the **CustomIdentityandJavaStandardTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_PROPERTIES="-Dweblogic.security.TrustKeyStore=JavaStandardTrust"
     ```

3. Run the **extractSSLArgs.sh** bash shell script **extractSSLArgs.sh** which is available under the Calendar Server installed location: *Calendar_Server_Installedlocation*/**sbin**:

   ```
   sh ./extractSSLArgs.sh -u weblogic_admin_user -p weblogic_admin_user_password -l
   t3s://weblogic_server_host:SSL_port
   ```

   If the script is successfully run, it creates **.wls_sslargs** under the configuration directory of your Oracle WebLogic Server domain. You can verify the creation of **.wls_sslargs** by navigating to the *Weblogic_Domain*/**config** directory.

Running the Script to Repair Keystore Information

If you modify keystore settings of the in-production WebLogic Server on which Calendar Server is deployed, you must update the previously supplied SSL Keystore details on Calendar Server. By using the **extractSSLArgs.sh -r** option, you can update the keystore information.

1. Ensure that WebLogic Server Keystore and SSL configurations are modified and you have the latest configuration details.
2. Open a new terminal and prepare the terminal by sourcing the **setDomainEnv.sh** script of the Oracle WebLogic Server domain:

   ```
   cd Weblogic_Domain/bin

   source ./setDomainEnv.sh   OR  . ./setDomainEnv.sh
   ```

3. Set the **WLST_PROPERTIES** environment variable. See step 2 in "Running the Script for a Fresh Installation". Ensure to enter the latest keystore details.
4. Run the following script:

   ```
   sh ./extractSSLArgs.sh -u weblogic_admin_user -p weblogic_admin_user_password -l
   t3s://weblogic_server_host:SSL_port  -r davserver_runtime_dir
   ```

   For example,

   sh extractSSLArgs.sh -u weblogic -p weblogic123 -l t3s://myhost.example.com:7002 -r /var/opt/sun/comms/davserver

5. Restart WebLogic Server.

   Weblogic_Domain/**config/.wls_sslargs** and *davserver_runtime_dir*/**config/davadmin.properties** files are updated according to the Keystore modifications performed on WebLogic Server.

   > **Note:**
   >
   > *CalendarServer_home*/**config/** contains the **davadmin.properties** file. CLI **davadmin** utility of the product uses the **davadmin.properties** file. Therefore, any modifications performed on WebLogic Server for keystore settings must reflect in the **davadmin.properties** file.

Database Installation Scripts

Calendar Server ships with Perl scripts to prepare a new database installation. You can use these scripts instead of manually preparing a new database installation.

config-mysql Script

The **config-mysql** script prepares a new MySQL installation for use with Calendar Server.

> **Caution:**
>
> If you already have a running instance of MySQL server, read the following information carefully before using this script.

Syntax and Examples

```
config-mysql [options]
```

Table B-3 describes the **config-mysql** *options*:

Table B-3 config-mysql Options

Option	Description
--**help** | **-?**	Prints help.
--calendar | **-c**	Configures the calendar database.
--ischedule | **-i**	Configures the ischedule database.
--server | **-s**	Configures the MySQL server instance.
--user | **-u**	Creates the MySQL database user.

To set up a new MySQL Server instance, and the default back end and iSchedule back end:

```
config-mysql -s -u -c -i
```

To set up a MySQL Server instance on an additional host, and a new calendar back end:

```
config-mysql -s -u -c
```

To set up just a new calendar back end on any existing instance:

```
config-mysql -c
```

Running the config-mysql Script

The **config-mysql** script creates a minimal instance configuration in the **/etc/my.cnf** file for use with the Calendar Server database.

To run the **config-mysql** script:

1. Copy the **config-mysql** and **Util.pm** scripts from a machine where Calendar Server is installed to the machine where MySQL is installed.

   Both scripts are located in the *CalendarServer_home***/tools/unsupported/bin** directory.

2. On the machine where MySQL is installed, as **root**, launch the **config-mysql** script with options **-s**, **-u**, **-c**, and **-i**.

   The script creates a log file in the **/tmp** directory with a name such as **config-mysql_***YYYYMMDDHHMMSS***.log**.

3. If you receive a message that the script has found existing instance configuration in the **/etc/my.cnf** file, enter `yes` to overwrite the existing instance configuration, or `no` to exit.

4. Enter the instance data directory.

If you choose a directory that already exists, the script assumes that it belongs to an existing MySQL instance and that it might contain useful data. If that's not the case and you want to use that directory, remove the directory first.

5. Enter the MySQL root user password, and enter again to confirm.
6. Enter the db user, password (once more to confirm), calendar db name, and iSchedule db name.

   The script then outputs the list of tasks to be performed and asks you to confirm before proceeding.
7. Enter `y` to proceed.

   The script performs the following steps:

   - Stops the running MySQL instance.

     This step is run regardless of whether an instance is running. You might see the following message, which you can safely ignore:

     ```
     ERROR! MySQL server PID file could not be found!
     ```
   - Runs the **mysql_install_db** command to create the instance data directory specified in Step 3.
   - Creates a minimal **/etc/my.cnf** file for the instance.
   - Copies the **mysql.server** script to the system startup and shutdown directory.
   - Starts the MySQL server by using the configuration in the **/etc/my.cnf** file.
   - Uses the **mysqladmin** script to change the `root` password specified in Step 5.
   - Runs **mysql_secure_installation** script to secure the newly created MySQL instance.
   - When the script asks for the existing root password, use the one from Step 5 and answer `no` to the question "Change the root password? [Y/n]."

     One of the tasks in the **mysql_secure_installation** script is to change the `root` password (because a fresh instance has a blank `root` password).
   - The last step is for the script to create the calendar database, iSchedule database, and database user by using the **mysql** tool.

config-oracle Script

The **config-oracle** script prepares a running Oracle Database instance for use with Calendar Server. The **config-oracle** script is supported by Oracle Database 11g Release 2 and Oracle Database 12c, not pluggable (non-CDB).

> **Note:**
>
> You cannot use the **config-oracle** script for an Oracle Database 12c container database (that is, one that uses a pluggable database). Instead, you must manually prepare an Oracle Database 12c container database for use with Calendar Server. See "Preparing Oracle Database 12c Container Database" for more information.

Before running this script, ensure that you have run the **oraenv** or **coraenv** script. The script creates a log file in the **/tmp** directory with a name such as **config-oracle**_*YYYYMMDDHHMMSS*`.log`.

Syntax and Examples

```
config-oracle [options]
```

Table B-4 describes the **config-oracle** *options*:

Table B-4 config-oracle Options

Option	Description
--**help** | **-?**	Prints help.
--calendar | **-c**	Configures the calendar database.
--ischedule | **-i**	Configures the iSchedule database.

To set up the default back end and iSchedule database:

```
config-oracle -c -i
```

To set up an additional calendar back end:

```
config-oracle -c
```

Running the config-oracle Script

To run the **config-oracle** script:

1. Copy the **config-oracle** and **Util.pm** scripts from a machine where Calendar Server is installed to the machine where Oracle Database is installed.

   Both scripts are located in the *CalendarServer_home***/tools/unsupported/bin** directory.
2. On the machine where Oracle Database is installed, as **root**, enter the following command:

   ```
   config-oracle -c -i
   ```
3. Enter the Oracle SYS user password.
4. Enter the calendar db user name and password, and iSchedule db user name and password.

   The script outputs the list of tasks to be performed.
5. Enter **`Y`** to proceed.

   The script creates the calendar and iSchedule database users and schemas.

populate-davuniqueid Script

Use the **populate-davuniqueid** script if you initially selected the **nsUniqueId** attribute as the unique identifier for your Calendar Server deployment. The **populate-davuniqueid** script populates calendar users and resources in LDAP with the **davUniqueId** schema element, introduced in Calendar Server 7 Update 3. It copies the value of **nsUniqueId** to **davUniqueId**, thus preserving references in the calendar database. The **davUniqueId** attribute is subsequently used as the value for the Calendar Server **davcore.uriinfo.permanentuniqueid** configuration parameter. Calendar Server requires a unique identifier in the form of an LDAP attribute whose value is used to map each calendar entry (in the LDAP Directory Server) to a unique account in the calendar database (the MySQL Server or Oracle Database that stores Calendar Server data). The unique identifier links various entries from different database tables for a user and resource. You must use a unique identifier, and one that does not change, for user and resource entries stored in LDAP.

> **Note:**
>
> The problem with using the **nsUniqueId** attribute for this purpose is that if the user or resource LDAP entry is deleted and re-created in LDAP, the new entry has a different **nsUniqueId** value. See "Changing the User Unique Identifier" for more information on the problems with using **nsUniqueId**.

Syntax

```
populate-davuniqueid [ -h host ] [ -p port ] [ -D user ]
                     [ -w pass | -j passfile ] [ -b base ] [ -f filter ]
                     [ -o outfile ] [ -O ] [ -x ] [ -v ] [ -? ]
```

Options

Table B-5 describes the **populate-davuniqueid** options:

Table B-5 populate-davuniqueid Options

Option	Description	Default Value
-h *host*	Specifies the Directory Server host	No default value.
-p *port*	Specifies the Directory Server port	**389**
-D *user*	Specifies the Directory Server bind user	No default value.
[**-w** *pass* | **-j** *passfile*] (The **-j** option is preferred, as the password is not specified on the command line.)	Specifies either the Directory Server bind password or a password file storing the directory user password	No default value.
-b *base*	Specifies the Directory base to carry out the search	No default value.
-f *filter*	Specifies the LDAP search filter	**"(|(objectclass=inetorgperson)(objectclass=inetresource)(objectclass=groupofuniquenames)(objectclass=groupofurls)(objectclass=inetmailgroup))"**
-o *outfile*	Specifies the output file name	No default value.
-O	Specifies to add the **davEntity** object class to the LDAP entry	No default value.
-x	Automatically runs the **ldapmodify** command on the output file	No default value.
-v	Specifies verbose debugging	No default value.
-?	Displays the usage help text	No default value.

The **populate-davuniqueid** script prompts you to type all options with the exception of *port* and *filter*, if you do not specify them. If not specified, **port** has a default value of 389 and **filter** has the value **"(|(objectclass=icscalendaruser)(objectclass=icscalendarresource)(objectclass=groupofuniquenames)(objectclass=groupofurls)**

(objectclass=inetmailgroup))". All LDAP entries matching *filter*, starting from the search *base*, that have a missing **davUniqueId** field are output to the *outfile* file in the form of an LDAP modification operation. You can then examine the *outfile* content before running it through an **ldapmodify** command, Alternately, you can do so automatically by using the **-x** option.

populate-davuniqueid Examples

These examples show different scenarios for running the **populate-davuniqueid** script.

Adding davUniqueId to All Calendar Server Users

To add the **davuniqueid** attribute to all Calendar Server users:

1. Run the **populate-davuniqueid** script and output the changes to a file, **sample.txt**.

```
/opt/sun/comms/davserver/sbin/populate-davuniqueid -h host1.us.example.com -D
"cn=Directory Manager" -b o=isp -o sample.txt
Directory user bind password:

Please check the following information
Directory host: host1.us.example.com
Directory port: 389
Directory user bind DN: cn=Directory Manager
Directory user bind password: as specified
Directory search base: o=isp
Directory search filter: (|(objectclass=icscalendaruser)
(objectclass=icscalendarresource))
Output to: sample.txt
Add daventity objectclass: no
Load output LDIF automatically: no
Do you want to continue (y/n)?: y
sample.txt is created. Please examine content and run ldapmodify on it.
```

2. Examine the **sample.txt** file. It should resemble the following:

```
dn: uid=calmaster63, ou=People, o=us.example.com,o=isp
changetype: modify
add: davuniqueid
davuniqueid: e5cb6c08-dd5811e1-80f5ce3b-d63ea23a
```

3. As long as there is no error, run the **ldapmodify** command to add **davUniqueId** to all Oracle Communications Calendar Server users, or run the following command:

```
/opt/sun/comms/davserver/sbin/populate-davuniqueid -h host1.us.example.com -D
"cn=Directory Manager" -b o=isp -o sample.txt -x
Directory user bind password:
```

Adding the daventity Object Class and davuniqueid to All Calendar Server 6 Users

To add the **daventity** object class and **davuniqueid** attribute to all Calendar Server 6 users:

1. Run the **populate-davuniqueid** script and output the changes to a file, **test.txt**.

```
/opt/sun/comms/davserver/sbin/populate-davuniqueid -h host2.us.example.com -D
"cn=Directory Manager" -b o=isp -o test.txt -O
Directory user bind password:

Please check the following information
Directory host: host2.us.example.com
Directory port: 389
Directory user bind DN: cn=Directory Manager
Directory user bind password: as specified
Directory search base: o=isp
Directory search filter: (|(objectclass=icscalendaruser)
(objectclass=icscalendarresource))
Output to: test.txt
Add daventity objectclass: yes
Load output LDIF automatically: no

Do you want to continue (y/n)?: y
test.txt is created. Please examine content and run ldapmodify on it.
```

2. Examine the **test.txt** file. It should resemble the following:

```
dn: uid=user3,ou=People,o=domain3.com,o=isp
changetype: modify
add: objectclass
objectclass: daventity
-
add: davuniqueid
davuniqueid: ff22a401-e52011e1-80f5ce3b-d63ea23a
```

3. As long as there is no error, run the **ldapmodify** command to add **davEntity** and **davUniqueId** to all Calendar Server 6 users, or run the following command:

```
/opt/sun/comms/davserver/sbin/populate-davuniqueid -h host2.us.example.com -D
"cn=Directory Manager" -b o=isp -o test.txt -O -x
Directory user bind password:
```

C

comm_dssetup.pl Reference

This appendix provides information about the Oracle Communications Calendar Server **comm_dssetup.pl** script. You must prepare your Oracle Directory Server Enterprise Edition (Directory Server) hosts by running the **comm_dssetup.pl** before you install and configure Calendar Server.

About the comm_dssetup.pl Script

This section provides information you need to understand before running the **comm_dssetup.pl** script.

The **comm_dssetup.pl** script performs the following steps:

1. Prompts you for your deployment's Directory Server and schema information.

   For a list of the specific information this step requests, see "Information Needed to Run the comm_dssetup.pl Script".

2. Generates a shell script and LDIF file from the information that you supply that is used to modify the Directory Server LDAP.

3. Runs the generated shell script and modifies your Directory Server.

At the end of each step, the **comm_dssetup.pl** script prompts you to continue. No changes are made to the Directory Server LDAP until the last step.

Directory Server Considerations for the comm_dssetup.pl Script

When running the **comm_dssetup.pl** script, consider the following points.

- **comm_dssetup.pl** configures local Directory Server instances, and thus you must:
  - Install the **comm_dssetup.pl** script on every host on which a Directory Server instance resides.
  - Run the **comm_dssetup.pl** script on the same host as your Directory Server. The tool runs locally for a specific instance (specified by path of Directory Server or path of instance).
- You can run the **comm_dssetup.pl** script against any Directory Server instance on the local host. If you have multiple Directory Information Trees (DITs) on one host, you can maintain and update one installation of **comm_dssetup.pl**, and apply it to every Directory Server instance on the host.
- **comm_dssetup.pl** must configure every Directory Server instance for the same DIT. This assumes that:
  - A Directory Server has already been installed, configured, and is running before you launch the **comm_dssetup.pl** script.
  - When adding an additional Directory Server host (such as a replica), at a future date, you must run the **comm_dssetup.pl** script against it, too.
- If you have customized your Directory Server, the following considerations might apply:

- If you have indexed some attributes, you might have to reindex those attributes after running the **comm_dssetup.pl** script.
- If you have added other LDIF files (schema definitions), they should not be affected, so no action should be necessary. However, back up your custom schema definition files before running the **comm_dssetup.pl** script.

  The **comm_dssetup.pl** script backs up old schema files to the **/var/tmp/dssetup_timestamp/save** directory.
- For all Directory Server customizations, including the first two just listed, stop the **comm_dssetup.pl** script after it generates the script and before it actually updates the LDAP directory. Then inspect the script to evaluate how its proposed actions affect your LDAP directory. Take whatever actions you think necessary to protect your customizations before running the script against your Directory Server.

Information Needed to Run the comm_dssetup.pl Script

Table C-1 describes the information about your deployment that you need to gather before running the **comm_dssetup.pl** script.

Table C-1 comm_dssetup.pl Information

Information Item Needed	**Default Value**
Directory Server root path name	The default depends on the Directory Server version detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Which instance of Directory Server to use? (if more than one)	The default depends on the Directory Server version detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Directory Manager Distinguished Name (DN)	"**cn=Directory Manager**"
Directory Manager's Password	NA
Directory Server being used for user/group data? (yes), or configuration data only? (no)	**yes**
User and group root suffix (if yes to previous question)	The default depends on what is detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Schema version? (pick one of the following): • 1 - Schema 1 • 1.5 - Schema 2 Compatibility Mode • 2 - Schema 2 Native Mode For more information on how to choose a schema, see "About the comm_dssetup.pl Script Schema Choices". If you have one version of the schema installed and want to upgrade to a higher level, refer to *Sun Java System Communications Services 6 2005Q4 Schema Migration Guide* before running the script.	**2** However, if you run **comm_dssetup.pl** again, it defaults to the value that you chose the previous time.
If you choose Schema 1 or 1.5, you need a DC tree. If the DC tree does not yet exist, the **comm_dssetup.pl** script creates only the root suffix node, its does not create the rest of the DC tree. You must create the rest of your DC tree yourself.	**o=internet** However, if you run **comm_dssetup.pl** again, it defaults to the value that you chose the previous time.

About the Directory Server Root Path Name and Instance

The **comm_dssetup.pl** script prompts you for both the Directory Server root path and the Directory Server instance. The script then combines these two items into an absolute path name to the Directory Server instance. For example, if your Directory Server instance resides under the **/var/opt/sun/directory/slapd-varrius** directory, then you specify **/var/opt/sun/directory** for the Directory Server root path and **slapd-varrius** for the Directory Server instance.

The reason for having two **comm_dssetup.pl** prompts to specify one absolute path is historical. Prior to Directory Server 6, Directory Server had the concept of a "server root" under which all Directory Server instances (and the Directory Server binaries) resided. After Directory Server 6, the concept of the "server root" was removed. Directory Server instances (and the Directory Server binaries) do not all have to reside under a single umbrella "server root" directory.

About the comm_dssetup.pl Script Schema Choices

Calendar Server supports the following schema choices:

- LDAP Schema 2 native mode

  Corresponds to **comm_dssetup.pl** script schema version choice 2. This is the default for a fresh installation.
- LDAP Schema 1

  Corresponds to the **comm_dssetup.pl** script schema version choice 1.
- LDAP Schema 2 compatibility mode

  Corresponds to **comm_dssetup.pl** script schema version choice 1.5.

About LDAP Schema 2

LDAP Schema 2 is a set of provisioning definitions that describes the types of information that can be stored as entries by using the Directory Server LDAP.

The native mode uses search templates to search the Directory Server LDAP. Once the domain is found by using the domain search template, the user or group search templates are used to find a specific user or group.

You should use native mode if you are installing Calendar Server for the first time and you do not have other applications in your deployment that are dependent on a two-tree provisioning model.

> **Note:**
>
> If you have an existing deployment that uses Schema 1, and you want to integrate other Communications Suite products, you should migrate your directory to Schema 2. Refer to *Sun Java System Communications Services 6 2005Q4 Schema Migration Guide* for information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2.

About LDAP Schema 1

LDAP Schema 1 is a provisioning schema that consists of both an Organization Tree and a DC Tree. In Schema 1, when a search is conducted for user or group entries, it looks at the user's or group's domain node in the DC Tree and extracts the value of the **inetDomainBaseDN** attribute. This attribute holds a DN reference to the organization subtree containing the actual user or group entry.

About LDAP Schema 2 Compatibility Mode

Schema 2 compatibility mode is an interim mode between Schema 1 and Schema 2 native mode. Schema 2 compatibility mode supports both schemas and enables you to retain the existing two-tree design you already have.

Use Schema 2 Compatibility if you have existing applications that require Schema 1, but you also need functionality that requires Schema 2.

> **Note:**
>
> Schema 2 compatibility mode is provided as a convenience in migrating to the Schema 2 Native mode. Do not use Schema 2 compatibility mode as your final schema choice. The migration process from Schema 1 to Schema 2 compatibility mode and then finally to Schema 2 native mode is more complex that simply migrating from Schema 1 to Schema 2 native mode. See *Sun Java System Communications Services 6 2005Q4 Schema Migration Guide* for more information.

Attribute Indexes Created by the comm_dssetup.pl Script

Attribute indexes improve the performance of search algorithms. The **comm_dssetup.pl** script offers you the choice to index attributes.

Table C-2 lists all the attributes the **comm_dssetup.pl** script indexes, grouped by suffix category. It also lists the type of indexes created for each attribute. For more information about Directory Server indexing, see the Directory Server documentation at:

`http://docs.oracle.com/cd/E20295_01/index.htm`

Table C-2 Attributes Indexed by comm_dssetup.pl

Suffix	Attributes Indexed	Type of Indexes Added
User/Group	**mail**	**pres**, **eq**, **approx**, **sub**
User/Group	**mailAlternateAddress**	**pres**, **eq**, **approx**, **sub**
User/Group	**mailEquivalentAddress**	**pres**, **eq**, **approx**, **sub**
User/Group	**mailUserStatus**	**pres**, **eq**
User/Group	**member**	**eq**
User/Group	**ou**	**pres**
User/Group	**cosspecifier**	**pres**
User/Group	**groupid**	**pres**, **eq**, **sub**

Table C-2 (Cont.) Attributes Indexed by comm_dssetup.pl

Suffix	Attributes Indexed	Type of Indexes Added
User/Group	**icsCalendar**	**pres**, **eq**, **approx**, **sub**
User/Group	**icsCalendarOwned**	**pres**, **eq**, **approx**, **sub**
User/Group	**uniqueMember**	**eq**
User/Group	**memberOf**	**eq**, **sub**
User/Group	**cn**	**eq**
User/Group	**mgrpUniqueId**	**eq**
User/Group	**deleted**	**pres**, **eq**
User/Group	**davuniqueid**	**pres**, **eq**
User/Group	**inetCos**	**eq**
User/Group (additional for Schema 2)	**inetDomainBaseDN**	**pres**, **eq**
User/Group (additional for Schema 2)	**sunPreferredDomain**	**pres**, **eq**
User/Group (additional for Schema 2)	**associatedDomain**	**pres**, **eq**
User/Group (additional for Schema 2)	**o**	**pres**, **eq**
User/Group (additional for Schema 2)	**mailDomainStatus**	**pres**, **eq**
User/Group (additional for Schema 2)	**sunOrganizationAlias**	**pres**, **eq**
DC Tree (for Schema 1)	**inetDomainBaseDN**	**pres**, **eq**
DC Tree (for Schema 1)	**mailDomainStatus**	**pres**, **eq**
DC Tree (for Schema 1)	**inetCanonicalDomainName**	**pres**, **eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfManagedGroup**	**pres**, **eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfPAB**	**pres**, **eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfPABGroup**	**pres**,**eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**un**	**eq**
New PAB (**o=PiServerDb**)	**displayname**	**pres**, **eq**, **sub**
New PAB (**o=PiServerDb**)	**MemberOfPiBook**	**eq**
New PAB (o=PiServerDb)	**MemberofPiGroup**	**eq**
o=mlusers for future mailserv feature	**mail**	**eq**
o=mlusers for future mailserv feature	**mlsubListIdentifier**	**eq**
o=mlusers for future mailserv feature	**mlsubMail**	**eq**

To add additional indexes on your own, see the Directory Server documentation.

Running the comm_dssetup.pl Script

You can run the **comm_dssetup.pl** script in either interactive or silent mode. Interactive mode is described in "Running the comm_dssetup.pl Script in Interactive Mode".

Running the comm_dssetup.pl Script in Silent Mode

To run the **comm_dssetup.pl** script in silent mode:

1. On the host where Directory Server is installed, log in as or become the superuser (**root**).
2. Start Directory Server, if necessary.
3. Change to the directory where you installed or copied the Directory Server Setup **comm_dssetup.pl** script.
4. Run the script followed by the silent mode options.

   All options are required. For more information, see "Silent Mode Options".

   ```
   /usr/bin/perl comm_dssetup.pl
   [ -i yes | no ] [ -R yes | no ] [ -c DirectoryServerRoot ]
   [ -d DirectoryInstance ] [ -r DCTreeSuffix ]
   [ -u UserGroupSuffix ] [ -s yes | no ] [ -D DirectoryManagerDN ]
   [ -j DirectoryManagerPasswordFile ] [ -b yes | no ]
   [ -t 1 | 1.5 | 2 ] [ -m yes | no ] [-S PathtoSchemaFiles ]
   ```

   The script creates the following LDIF file and shell script to update the LDAP indexes and schema:

   - **/var/tmp/***dssetup_timestamp***/dssetup.ldif**
   - **/var/tmp/***dssetup_timestamp***/dssetup.sh**

5. If you answered `no` to the **-R** and **-m** options, you must manually run the **comm_dssetup.sh** script that was created.

   If you answered `yes` to the **-R** and **-m** options, the **dssetup.sh script** is run automatically.

Silent Mode Options

Table C-3 table describes the **comm_dssetup.pl** silent mode options.

Table C-3 comm_dssetup.pl Silent Mode Options

Option and Argument	Description
-i yes |**no**	Specifies whether to configure new indexes. **yes** - Add new Directory Server indexes. **no** - Do not add indexes.
-R yes | **no**	Specifies whether to reindex automatically. **yes** - Reindex without prompting the user. **no** - Do not reindex without prompting the user. The **-m** option must also be specified for **yes** for the **-R** option to take effect.
-c *DirectoryServerRoot*	Specifies the Directory Server root path, for example, **/var/opt/sun/directory**.
-d *DirectoryInstance*	Specifies the Directory Server instance subdirectory under the Directory Server root path, for example, **slapd-varrius**.
-r *DCTreeSuffix*	Specifies the DC tree root suffix (for Schema 1 and Schema 2 compatibility modes only), for example, **o=internet**.
-u *UserGroupSuffix*	Specifies the user and group root suffix, for example, **o=usergroup**.

Table C-3 (Cont.) comm_dssetup.pl Silent Mode Options

Option and Argument	Description
-s yes | no	Specifies whether to update the schema. **yes** - Update the schema. **no** - Do not update schema.
-D *DirectoryManagerDN*	Specifies the Directory Manager Distinguished Name (DN), for example, **"cn=Directory Manager"**. The value must be enclosed by double quotation marks (" ") to enable the **comm_dssetup.pl** script to interpret a value with a space correctly.
-j *DirectoryManagerPasswordFile*	Specifies the file containing the Directory Manager DN password.
-b yes | **no**	Specifies to use this Directory Server for users and groups. **yes** - Use this directory to store both configuration and user group data. **no** - Use this directory to store only configuration data.
-t 1 | **1.5** | **2**	Specifies the schema version.
-m yes | **no**	Specifies whether to modify the Directory Server. **yes** - Modify the Directory Server without prompting the user. **no** - Do not modify the Directory Server without prompting the user.
-S *PathtoSchemaFiles*	Specifies the path to the directory where the schema files are located for example, **./schema**.

D

rundssetup Reference

Before Configuring UCS Products, you must prepare your Directory Server host. This is done by running the directory server setup script (**rundssetup**) against the directory Server Instance.

This appendix provides information about the **rundssetup** script.

Calendar Server release 8.0.0.7.0 is certified using Oracle Unified Directory. DS Setup 6.4.0.30.0 is required for Oracle Unified Directory support.

DS Setup version 6.4.0.30.0 uses **rundssetup** script which is an enhanced version of **comm_dssetup.pl** script available in earlier versions of DS Setup. This appendix provides information on **rundssetup** script usage to prepare Oracle Unified Directory.

Downloading and Installing DS Setup

1. Download the DS Setup from: https://support.oracle.com

   The **rundssetup** script is available in the same software package as the Calendar Server software.

2. Copy the directory server setup ZIP file to a temporary directory on your directory server hosts and extract the files.
3. Log on to the directory server host machine as the superuser (root).
4. Change to the directory where you extracted the DS Setup.
5. Install the DS Setup using:

   ```
   ./commpkg install
   ```

6. Select **Comms DSsetup** from the list of applications to install and proceed with the installation.

   See commpkg Reference for more information about the **commpkg** command.

About the rundssetup Script

This section provides the information you need to understand before running the **rundssetup** script.

The **rundssetup** script performs the following steps:

1. Prompts you for your deployment's Directory Server and schema information.

   For a list of the specific information this step requests, see Information Needed to Run the rundssetup Script

2. Generates a shell script and LDIF file from the information that you supply that is used to modify the Directory Server LDAP.
3. Runs the generated shell script and modifies your Directory Server.

At the end of each step, the **rundssetup** script prompts you to continue. No changes are made to the Directory Server LDAP until the last step.

Running rundssetup script

Before running DS Setup, make sure the following are installed:

- Directory Server (Oracle Unified Directory)
- Python 2.7 and above

To install Oracle Unified Directory, see Install the Oracle Unified Directory Software

For setting up the directory server, see Setting Up Oracle Unified Directory as a Directory Server

Make sure a Directory Server instance is already created and is started. The **rundssetup** script can be run in interactive mode or silent mode

To run the script for preparing OUD, run the command:

```
python rundssetup --dsType OUD
```

Answer the command-line prompts.

> **Note:**
>
> You can use either LDAP Schema 2 or Schema 1. Schema 2 is recommended.

To run **rundssetup** script in silent mode, see "Running **rundssetup** in silent mode".

If the directory server is already installed at your site, users have already been provisioned. If you have just installed the directory server at your site, then you need to provision users. For information about provisioning users and schema, see Communications Suite Schema Reference.

Directory Server Considerations for the rundssetup Script

When running the **rundssetup** script, consider the following points.

- **rundssetup** configures local Directory Server instances, and thus you must:
  - Install the **rundssetup** script on every host on which a Directory Server instance resides.
  - Run the **rundssetup** script on the same host as your Directory Server. The tool runs locally for a specific instance (specified by path of Directory Server or path of instance).
- You can run the **rundssetup** script against any Directory Server instance on the local host. If you have multiple Directory Information Trees (DITs) on one host, you can maintain and update one installation of **rundssetup**, and apply it to every Directory Server instance on the host.
- **rundssetup** must configure every Directory Server instance for the same DIT. This assumes that:
  - A Directory Server has already been installed, configured, and is running before you launch the **rundssetup** script.

  - When adding an additional Directory Server host (such as a replica), at a future date, you must run the **rundssetup** script against it, too.
- If you have customized your Directory Server, the following considerations might apply:
  - If you have indexed some attributes, you might have to reindex those attributes after running the **rundssetup** script.
  - If you have added other LDIF files (schema definitions), they should not be affected, so no action should be necessary. However, back up your custom schema definition files before running the **rundssetup** script. The **rundssetup** script backs up old schema files to the **/var/tmp/dssetup_timestamp/save** directory.
  - For all Directory Server customizations, including the first two just listed, stop the **rundssetup** script after it generates the script and before it actually updates the LDAP directory. Then inspect the script to evaluate how its proposed actions affect your LDAP directory. Take whatever actions you think necessary to protect your customizations before running the script against your Directory Server.

Information Needed to Run the rundssetup Script

Table D-1 describes the information about your deployment that you need to gather before running the **rundssetup** script.

Table D-1 rundssetup Information

Information Item Needed	Default Value
Directory Server's Instance Path	The instance of Directory Server to be used (if more than one). (Absolute path of the Directory Server instance where it is running). The script displays an example of instance path of Directory Server, depending on the Directory Server type selected. The **rundssetup** script does attempt to heuristically determine the default. By default, the dstype is ODSEE. Example for Directory Server instance path when dstype is OUD: **/opt/oracle/Oracle/Middleware/asinst_1**
User and group root suffix	The default depends on what is detected. The **rundssetup** script does attempt to heuristically determine the value. For example, o=usergroup
Schema version? (pick one of the following): • 1 - Schema 1 • 1.5 - Schema 2 Compatibility Mode • 2 - Schema 2 Native Mode For more information on how to choose a schema, see "About the rundssetup Script Schema Choices". For new deployments , Schema 2 is preferred.	2
Do you want to update the schema files	Yes
Do you want to configure new indexes	Yes
Do you want to Reindex the new indexes now	Yes

Table D-1 (Cont.) rundssetup Information

Information Item Needed	Default Value
DC Tree base suffix (This option is not prompted when schema type is chosen as 2). If you choose Schema 1 or 1.5, you need a DC tree. If the DC tree does not yet exist, the **rundssetup** script creates only the root suffix node, its does not create the rest of the DC tree. You must create the rest of your DC tree yourself.	o=internet However, if you run **rundssetup** again, it defaults to the value that you chose the previous time.

About the rundssetup Script Schema Choices

Attribute Indexes Created by the rundssetup Script

Attribute indexes improve the performance of search algorithms. The **rundssetup** script offers you the choice to index attributes.

Table D-2 lists all the attributes the **rundssetup** script indexes, grouped by suffix category. It also lists the type of indexes created for each attribute. For more information about Directory Server indexing, see the Directory Server documentation at: https://docs.oracle.com/cd/E52734_01/oud/OUDAG/indexing.htm#OUDAG00048

Attribute indexes improve the performance of search algorithms. The **rundssetup** script offers you the choice to index attributes.

Table D-2 Attributes Indexed by rundssetup

Suffix	Attributes Indexed	Types of Indexes Added
User/Group (schema 1 & 2)	mail	'presence','equality','approximate','substring'
N/A	mailAlternateAddress	'presence','equality','approximate','substring'
N/A	mailEquivalentAddress	'presence','equality','approximate','substring'
N/A	mailUserStatus	'presence','equality'
N/A	member	'equality'
N/A	ou	'presence'
N/A	groupid	'presence','equality','substring'
N/A	uniquemember	'equality'
N/A	memberOf	'substring','equality'
N/A	cn	'equality'
N/A	mgrpUniqueId	'equality'
N/A	deleted	'presence','equality'
N/A	davuniqueid	'presence','equality'
N/A	inetCos	'equality'
User/Group (schema 2)	inetDomainBaseDN	'presence','equality'

Table D-2 (Cont.) Attributes Indexed by rundssetup

Suffix	Attributes Indexed	Types of Indexes Added
N/A	sunPreferredDomain	'presence','equality'
N/A	associatedDomain	'presence','equality'
N/A	o	'presence','equality'
N/A	mailDomainStatus	'presence','equality'
N/A	sunOrganizationAlias	'presence','equality'
DC Tree (Schema 1)	inetDomainBaseDN	'presence','equality'
(o=internet)	mailDomainStatus	'presence','equality'
N/A	inetCanonicalDomainName	'presence','equality'
New PAB (o=PiServerDb)	displayname	'presence','equality','substring'
N/A	memberOfPIBook	'equality'
N/A	memberOfPIGroup	'equality'
o=mlusers	mail	'equality'
N/A	mlsubListIdentifier	'equality'
N/A	mlsubMail	'equality'

DS Setup command line options

Table D-3 describes **rundssetup** command line options. All options are not mandatory. The options not specified are picked from default values. But if options are provided, they override the default ones.

Table D-3 DS Setup Command Line Options

Option and Argument	Description
-h, --help	Shows the help message and exits
--version	Show program's version number and exit
--debug	Turns on debugging output
--verbose	Verbose output
-D BINDDN, --bindDN BINDDN	DS bind DN credential, e.g. "cn=Directory Manager"
-j PASSWDFILE, --bindPasswordFile PASSWDFILE	The file containing DS bind DN password, e.g. "mypasswdfile"
-i {yes,no}, --addIndex {yes,no}	Add new indexes yes/no, e.g. "yes"
-R {yes,no}, --reIndex {yes,no}	Run reindexing
-d INSTLOC, --instanceLocation INSTLOC	Location of DS instance, e.g. "/oracle/Oracle/Middleware/asinst_1"
-r DCTREE, --dctree DCTREE	DC tree suffix, e.g. "o=internet"
-u UGSUFFIX, --ugtree UGSUFFIX	User/Group tree suffix, e.g. "o=usergroup"
-s {yes,no}, --updateSchema {yes,no}	Whether to update schema (yes/no), e.g. "yes"
-t {1,1.5,2}, --schemaType {1,1.5,2}	The schema type (1, 1.5, or 2), e.g. "2"

Table D-3 (Cont.) DS Setup Command Line Options

Option and Argument	Description
-m {yes,no}, --modifyDS {yes,no}	Whether to modify the Directory Server (yes/no), e.g."yes"
-f {yes,no}, --force {yes,no}	Force the application of this version of dssetup, even if the same version or later has been applied before
--silent SILENTFILE	Run silently, taking input from SILENTFILE and the command line. Command line arguments override entries in SILENTFILE. Specify NONE for the SILENTFILE if you want silent mode but with no silent file
--dsType {OUD,DSEE}	The Directory Server type, e.g. "OUD"
--createSuffixes {yes,no}	Power User switch - whether to create suffixes (yes/no), e.g. "yes", default is yes
--createSuffixDN {yes,no}	Power user switch (OUD only) - whether to create the DN associated with the suffix (yes/no). If --createSuffixes is no, then this switch is ignored (i.e. will be "no"), e.g. "yes", default is yes
--createMLusersSuffix {yes,no}	Power user switch - whether to create the o=mlusers suffix (yes/no), e.g. "yes", default is yes
--createPiServerDbSuffix {yes,no}	Power user switch - whether to create the o=mlusers suffix (yes/no), e.g. "yes", default is yes

Running the rundssetup Script in Silent Mode

To run the **rundssetup** script in silent mode:

1. On the host where Directory Server is installed, log in as or become the superuser (root).
2. Start Directory Server, if necessary.
3. Change to the directory where you installed or copied the Directory Server Setup **rundssetup** script.
4. Run the script followed by the silent mode options.

   For more information, see "Silent Mode Options".

   ```
   rundssetup [-h] [--version] [--debug] [--verbose] [-D BINDDN]

   [-j PASSWDFILE] [-i {yes,no}] [-R {yes,no}] [-d INSTLOC]

   [-r DCTREE] [-u UGSUFFIX] [-s {yes,no}] [-t {1,1.5,2}]

   [-m {yes,no}] [-f {yes,no}] [--silent SILENTFILE]

   [--dsType {OUD,DSEE}] [--createSuffixes {yes,no}]

   [--createSuffixDN {yes,no}] [--createMLusersSuffix {yes,no}]

   [--createPiServerDbSuffix {yes,no}]
   ```

   The script creates the following LDIF file and shell script to update the LDAPindexes and schema:

- **/var/tmp/dssetup_timestamp/dssetup.ldif**
- **/var/tmp/dssetup_timestamp/dssetup.sh**

5. If you answered no to the `-R` and `-m` options, you need to manually run the dssetup.sh script that was created. If you answered yes to the `-R` and `-m` options, the **dssetup.sh** script is run automatically.

   For example:

   schema 1:

   ```
   rundssetup -D "cn=Directory Manager" -j /tmp/ds_pass -i yes -R yes -d /oracle/Oracle/
   Middleware/asinst_1 -r "o=internet" -u "o=usergroup" -s yes -t 1 -m yes -f yes --
   silent=NONE --dsType=OUD --createSuffixes yes --createSuffixDN no --
   createMLusersSuffix yes --createPiServerDbSuffix yes
   ```

   schema 2:

   ```
   rundssetup -D "cn=Directory Manager" -j /tmp/ds_pass -i yes -R yes -d /oracle/Oracle/
   Middleware/asinst_1 -u o=usergroup -s yes -t 2 -m yes -f yes --silent=NONE --
   dsType=OUD --createSuffixes yes --createSuffixDN no --createMLusersSuffix yes --
   createPiServerDbSuffix yes
   ```

Silent Mode Options

Following are the options supported for running in silent mode:

```
rundssetup [-h] [--version] [--debug] [--verbose] [-D BINDDN]

[-j PASSWDFILE] [-i {yes,no}] [-R {yes,no}] [-d INSTLOC]

[-r DCTREE] [-u UGSUFFIX] [-s {yes,no}] [-t {1,1.5,2}]

[-m {yes,no}] [-f {yes,no}] [--silent SILENTFILE]

[--dsType {OUD,DSEE}] [--createSuffixes {yes,no}]

[--createSuffixDN {yes,no}] [--createMLusersSuffix {yes,no}]

[--createPiServerDbSuffix {yes,no}]
```

Explanation of the above options can be seen by running:

```
./rundssetup --help
```

E

ODSEE to OUD Migration

This appendix provides information about migrating from ODSEE to OUD.

This topic outlines the steps for migrating ODSEE11g deployment having DSsetup version (6.4.0.27.0 or above) to the OUD12c with DSsetup version 6.4.0.30.0. For general guidelines on transitioning from ODSEE to OUD, see the links provided under the "References" section .

The example dealt with in this document uses the tightly coupled co-existence migration strategy and shows setting up three servers listed below.

- ODSEE (skip if it already exists)
- OUD
- OUD replication gateway

For the example in this document, all these servers are set up on a single host and hence non-standard ports will be used by these servers.

> **Note:**
>
> If you already have ODSEE11g with DSsetup version (6.4.0.27.0 or above), then skip steps 1, 2, and 3 in Migrating ODSEE Deployment to OUD.
>
> References to Hostname of this example machine have been masked as "HOSTNAME" throughout this document. Use FQDN of your machine for such references.

Assuming that the ODSEE11g deployment with DSsetup(6.4.0.27.0 or above) exists, along with the directory data, follow these steps:

1. Setting ODSEE password compatibility to DS mode
2. Setup an empty OUD instance
3. Analyzing the ODSEE data (export ODSEE data to do the analysis)
4. Migrating ODSEE schema to OUD
5. Migrating ODSEE configuration to OUD
6. Enable ODSEE replication
7. Setup the OUD replication gateway
8. Apply DSsetup 6.4.0.30.0 to update the schema on OUD
9. Export ODSEE data
10. Run dsreplication pre-external-initialization
11. Import ODSEE data into OUD
12. Run dsreplication post-external-initialization

The keys are:

- Must use an empty OUD instance
- Must use a single OUD instance (not replicated)
- Must set up the OUD replication gateway before exporting ODSEE data for import into OUD
- Must use the switches to set up the OUD replication gateway
- ODSEE must have password compatibility in DS6-mode
- ODSEE must have replication enabled

References

This topic outlines the extra reference to see while transitioning to OUD.

Transition to OUD (Technical Brief and Guide)

- Transitioning from Oracle Directory Server Enterprise Edition to Oracle Unified Directory
- Oracle® Fusion Middleware Transition Guide for Oracle Unified Directory

More Information on Migration from ODSEE to OUD

This section contains resources with more online resources on the migration from ODSEE to OUD:

- Simple OUD 11g Migration Example Using Replication Gateway and "ds2oud"
- How to Install and Configure Standalone Replication Gateway (RGW) Instances With Standalone OUD Instances and ODSEE Instances

OUD 12c Documentation Library

- Planning Oracle Unified Directory Installation
- Performing OUD 12.2.1.4 Silent Installation
- Setting up OUD 12.2.1.4 as Directory Server

ODSEE 11.1.7 Documentation Library

- Oracle Directory Server Enterprise Edition

About Migration Data Cleanup and Issues

This topic outlines the process of migration data cleanup and the issues that you might face during the process.

Migration Steps Depending on Product-Specific Data

The entire process of migration depends on the UCS Products used in the deployment and thereby the type of data residing in the ODSEE. The steps and issues covered in this topic will therefore be subjective to the sample ODSEE data used to begin this migration scenario. For each UCS Product, you must pay special attention to product-specific migration issues that might be occurring during certain steps described here, in particular at step 6 "Execute ds2oud –diagnose" and step 8 "Analyze ODSEE data" in Migrating ODSEE Deployment to OUD.

Migrating Schema and Indexes

When analyzing the ODSEE data, several issues will be flagged by "ds2oud" tool regarding schema issues. These schema issues fall under the responsibility of DSsetup. The migration step shown in this document- where DSsetup is run against OUD - should fix all schema issues. The indexes removal step with the sample data is also shown in this document. If you encounter any issues with respect to schema or indexes, please refer to the links provided under the respective section or consult Oracle support for further assistance.

Objectclass and Attributes Cleanup Issues

When ODSEE data is diagnosed against OUD Schema, there may be issues shown for unsupported Objectclass and Attributes, due to schema cleanup. Refer to the following UCS Schema Reference guides to know about the objectclass and attributes support:

- UCS 8.0 Schema Reference guide (The deprecated objectclass/attributes flagged as incompatible while running ds2oud step can be found here)
- UCS 8.1 Schema Reference guide (For more information about the various objectclasses and attributes, refer to this guide.)

For UCS Product-specific support or issues, refer to the respective product documentation or consult Oracle support for further assistance.

Prerequisites

The following are the prerequisites for the process of migration:

- ODSEE must be version 11.1.1.7.0 or greater
- ODSEE password compatibility must be set to DS6 mode
- Create the file **/tmp/passwd** containing your password
- DSsetup version 6.4.0.27.0 or above (to be applied on ODSEE) (For the example in this document, 6.4.0.29.0 is used)
- DSsetup version 6.4.0.30.0 (to be applied on OUD)
- OUD version 11g or 12c (OUD 12.2.1.4 has been certified)

Ports Used in the Migration Example

In this example deployment, the following ports are used for servers setup on a single machine:

Table E-1 Ports Used in the Migration Example

N/A	LDAP	LDAPS	ADMIN	SYNC
OUD DS	1389	1636	1444	1989
OUD Repl GW	1390	1637	1445	N/A
ODSEE DS	1393	1640	N/A	N/A

Migrating ODSEE Deployment to OUD

To migrate from ODSEE deployment to OUD, follow these steps:

1. **Installing ODSEE and creating an instance**

   If you already have ODSEE, you may skip this step.

   a. **Install ODSEE bits**

      - **cd /opt**
      - **unzip -q /export/ODSEE_ZIP_Distribution/sun-dsee7.zip**

      Sample Session

      ```
      # cd /opt

      # unzip -q /export/ODSEE_ZIP_Distribution/sun-dsee7.zip
      ```

   b. **Create ODSEE instance**

      - **/opt/dsee7/bin/dsadm create –port 1393 –secure-port 1640 –pwd-file /tmp/passwd /var/opt/sun/directory/ds7**
      - **/opt/dsee7/bin/dsadm start /var/opt/sun/directory/ds7**

      Sample Session

      ```
       # /opt/dsee7/bin/dsadm create --port 1393 --secure-port 1640 --pwd-
      file /tmp/passwd /var/opt/sun/directory/ds7

       Use command 'dsadm start '/var/opt/sun/directory/ds7'' to start the
      instance

       # /opt/dsee7/bin/dsadm start /var/opt/sun/directory/ds7

       Directory Server instance '/var/opt/sun/directory/ds7' started:
      pid=3295
      ```

2. **Install and run DSsetup 6.4.0.29.0**

   If you already have ODSEE with DSsetup 6.4.0.27.0 or above, then you may skip this step. In this example, DSsetup 6.4.0.29.0 is being installed for the ODSEE installed in step 1.

   a. Run **commpkg install** to install DSsetup 6.4.0.29.0.

   b. Run **/opt/sun/comms/dssetup/sbin/comm_dssetup.pl**.

   Sample Summary

   Here is a sample summary from the **comm_dssetup.pl**.

   ```
   Server Root                  : /var/opt/sun/directory

   Server Instance              : ds7

   Users/Groups Directory       : yes

   Update Schema                : yes
   ```

```
Schema Type                 : 2

DC Root                     : o=usergroup

User/Group Root             : o=usergroup

Add New Indexes             : yes

ReIndex New Indexes Now     : yes

Directory Manager DN        : cn=Directory Manager
```

3. **UCS Products setup with ODSEE and Provisioning**

   If you already have UCS products configured to backend ODSEE (having DSsetup 6.4.0.27.0 or above), along with existing domains/users/groups provisioned, then you may skip this step. At this stage, you can install and configure any required UCS products pointing to the ODSEE setup above. Refer to the respective product documentation for configuring UCS products. Then, provision the domains/users/groups required in ODSEE (populating ODSEE with data). If you have this data already in a valid LDIF format, then you may populate it into ODSEE as shown in the example below:

   Example: Populated ODSEE with data

   ```
   ldapmodify -D 'cn=Directory Manager' -j /tmp/passwd -h <HOSTNAME> -p 1393 -a -f /
   shared/resources/ucs_data.ldif
   ```

   As another example, shown below is the basic setup of Messaging Server(MS) product and also a sample "testuser1" created:

   - Unzip the Messaging Server ZIP file downloaded from MOS.
   - **./commpkg install**
   - **/opt/sun/comms/messaging64/bin/configure –ldapport=1393**
   - **/opt/sun/comms/messaging64/lib/inetuser create -D 'cn=Directory Manager' -j /tmp/passwd testuser1**

   > **Note:**
   >
   > If you installed a UCS product whose version does not support OUD yet, then the next steps are to perform specific steps for that product and then upgrade that product to a version that supports OUD. Refer to UCS Product documentation.

4. **Change ODSEE password compatibility to DS6 mode**

   Run the following commands:

   ```
   /opt/dsee7/bin/dsconf pwd-compat -port 1393 -accept-cert -user-dn 'cn=Directory
   Manager' -pwd-file /tmp/passwd to-DS6-migration-mode

   /opt/dsee7/bin/dsconf pwd-compat -port 1393 -accept-cert -user-dn 'cn=Directory
   Manager' -pwd-file /tmp/passwd to-DS6-mode
   ```

   #### Sample session

   ```
   # /opt/dsee7/bin/dsconf pwd-compat --port 1393 --accept-cert --user-dn 'cn=Directory
   Manager' --pwd-file /tmp/passwd to-DS6-migration-mode
   ```

```
## Beginning password policy compatibility changes.

## Password policy compatibility changes finished.

 Task completed (slapd exit code: 0).

 # /opt/dsee7/bin/dsconf pwd-compat --port 1393 --accept-cert --user-dn
'cn=Directory Manager' --pwd-file /tmp/passwd to-DS6-mode

 ## Beginning password policy compatibility changes.

 ## Password policy compatibility changes finished.

 Task completed (slapd exit code: 0).
```

5. **Install OUD and setup OUD Instance**

   a. **Installation of OUD**

      In this document example, the OUD 12.2.1.4.0 has been installed in Standalone mode.

   b. **OUD Instance setup**

      /opt/oracle/Oracle/Middleware/Oracle_OUD1/oud-setup --cli --no-prompt --rootUserPasswordfile /tmp/passwd --ldapPort 1389 --ldapsPort 1636 --adminConnectorPort 1444 --generateSelfSignedCertificate

      Sample Session

```
 #/opt/oracle/Oracle/Middleware/Oracle_OUD1/oud-setup --cli --no-prompt --
rootUserPasswordfile /tmp/passwd --ldapPort 1389 --ldapsPort 1636 --
adminConnectorPort 1444 --generateSelfSignedCertificate

 Oracle Unified Directory 12.2.1.4.0

 Please wait while the setup program initializes...

 Creating instance directory /opt/oracle/Oracle/Middleware/asinst_1/OUD .....
Done.

 See /opt/oracle/Oracle/Middleware/asinst_1/OUD/logs/oud-setup for a detailed
log of this operation.



Configuring Directory Server ..... Done.

Configuring Certificates ..... Done.

Starting Directory Server ....... Done.



To see basic server configuration status and configuration you can launch /opt/
oracle/Oracle/Middleware/asinst_1/OUD/bin/status
```

6. Run `ds2oud -diagnose`

   This step diagnoses the ODSEE data for OUD migration problems, using "ds2oud".

   ds2oud -diagnose:

```
/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --diagnose --odseeBindDN
'cn=Directory Manager' --odseeHostname <HOSTNAME> --odseePort 1393 --
odseeBindPasswordFile /tmp/passwd --no-prompt
```

Sample Session

```
# /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --diagnose --odseeBindDN
'cn=Directory Manager' --odseeHostname <HOSTNAME> --odseePort 1393 --
odseeBindPasswordFile /tmp/passwd --no-prompt

 ********************************************************************************

 Diagnose ODSEE Server : <HOSTNAME>:1393

 ********************************************************************************

 <...output snipped...>

 ** Encrypted attributes

 No encrypted attributes are defined, no action is required
```

7. **Export ODSEE data to ldif**

run `dsconf export`

```
/opt/dsee7/bin/dsconf export --accept-cert --user-dn 'cn=Directory Manager' --pwd-
file /tmp/passwd -f opends-export -f output-not-folded -h <HOSTNAME> -p 1393
o=usergroup o=mlusers o=PiServerDb odsee-data.ldif
```

> **Note:**
>
> - `-f opends-export`: is used to have it suitable for import-ldif later on OUD side. However, do not use the data from this `run/step` for `import-ldif`. After the replication gateway is setup the data will be exported and that must be used for `import-ldif`.
> - Do not include `o=comms-config`.
> - `-f output-not-folder` option: This is to avoid line folding. If not given, export causes bigger lines to get folder into multiple lines - which leads to issues while doing search/replace in data cleanup steps later (if required to clean ODSEE ldif data)
> - `o=usergroup`: is the user/group suffix considered in this sample. Ensure to include your suffix accordingly.
> - `o=mlusers`: this is for MS mailing lists
> - The case above is schema 2 (see output of DSsetup run). For schema 1, add for example o=internet (typically DC tree).

Sample Session

```
# /opt/dsee7/bin/dsconf export --accept-cert --user-dn 'cn=Directory Manager' --pwd-
file /tmp/passwd -f opendsexport -f output-not-folded -h <HOSTNAME> -p 1393
o=usergroup o=mlusers o=PiServerDb odsee-data.ldif

 ## Beginning export of 'usergroupdb2'

 ## usergroupdb2: Start processing.

 ## usergroupdb2: Processed 123 entries (100%), 123.0 entries/sec average, 123
exported.
```

```
 ## Beginning export of 'mlusersdb2'

 ## mlusersdb2: Start processing.

 ## mlusersdb2: Processed 1 entries (100%), 1.0 entries/sec average, 1 exported.

 ## Beginning export of 'PiServerDbdb2'

 ## PiServerDbdb2: Start processing.

 ## PiServerDbdb2: Processed 36 entries (100%), 36.0 entries/sec average, 36
exported.

 ## Export finished.
```

8. **Analyze ODSEE data**
   1. Move **odsee-data.ldif** to an accessible location
      - `cp /var/opt/sun/directory/ds7/logs/odsee-data.ldif /tmp`
      - If it is on a different machine, set permissions (chmod the file) and then:
        - `scp <HOSTNAME>:/var/opt/sun/directory/ds7/logs/odsee-data.ldif /tmp`
   2. run `ds2oud -ldifDBFile`

      ```
      /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --ldifDBFile /tmp/odsee-
      data.ldif --userSchemaFile /opt/sun/comms/dssetup/lib/foranalysis-oud-schema.ldif
      ```

      If it shows any incompatible objectclass/attributes, then clean those up from the ODSEE LDIF data. Refer to the UCS Schema Reference guides provided in the topic "Objectclass and Attributes Cleanup Issues" to know such deprecated information. This run might also show any unsupported/invalid keywords. You must fix the LDIF data by replacing the invalid keywords with the suggested keywords provided in the output. See the sample session below:

      Sample Session

      ```
      ********************************************************************************
      * Diagnose ODSEE LDIF data file : /tmp/odsee-data.ldif
      ********************************************************************************

      Error validating data against OUD schema
      Entry : unknown
      org.opends.sdk.DecodeException: Entry o=usergroup read from LDIF starting at
      line 8 includes value "(target="ldap:///o=usergroup")(targetattr="*")(version
      3.0;acl "Contacts Server End User Administrator Proxy Rights -
      product=nabserver,schema 2 support,class=admin,num=1,version=1"; allow (proxy)
      roledn="ldap:///cn=Contacts End User Administrators Group, ou=Groups,
      o=usergroup";)" for attribute aci that is invalid according to the
      associated syntax: The provided Access Control Instruction (ACI) expression
      value "ldap:///cn=Contacts End User Administrators Group, ou=Groups,
      o=usergroup" is invalid because it contains the roledn keyword, which is not
      supported, replace it with the groupdn keyword
      ```

> **Note:**
>
> - ACIs in OUD do not accept empty spaces.
> - Following replacements were required during this run with our test data:
>   - Replaced **roledn** with **groupdn**
>   - Replaced **groupdnattr** with **groupdn**

9. **Install DSsetup 6.4.0.30.0 for OUD**

   Install DSsetup 6.4.0.30.0

   On the machine where OUD is residing, download DSsetup 6.4.0.30.0 and configure this DSsetup version with OUD.

   - Download DSsetup 6.4.0.30.0 and unzip the ZIP obtained.
   - Run **commpkg install**

   > **Note:**
   >
   > The Dssetup utility for OUD support cleans up the UCS schema, removing object classes and attributes that are no longer utilized by UCS components.
   >
   > The attributes not supported in OUD are:
   >
   > - icsAllowedServiceAccess
   > - icsCalendarOwned
   > - icsExtendedDomainPrefs
   > - icsFreeBusy
   > - icsSubscribed
   > - sunAllowMultipleServices
   > - icsAdministrator
   > - icscalendardwphost
   > - iPlanetPreferences

10. Run `DSsetup 6.4.0.30.0 on OUD` to install just the schema

    Run DSsetup 6.4.0.30.0 as shown below, to install just the schema on the OUD instance. It is important that this step be done prior to running migrateUserSchema, which migrates the ODSEE schema into OUD.

    > **Note:**
    >
    > If this step is not done, then the schema attribute such as "iplanet-am-managed-group" could show up twice in 99-user.ldif on the OUD side.

    rundssetup command:

```
bin/rundssetup --dsType=OUD \

  --instanceLocation /opt/oracle/Oracle/Middleware/asinst_1 \

  --bindPasswordFile /tmp/passwd \

  --updateSchema yes \

  --createSuffixes no \

  --silent NONE \

  --modifyDS yes
```

11. Process `ds2oud migrateUserSchema` (optional)

    This will migrate ODSEE schema into OUD. This is an optional step. It is not recommended to do this to see if entries have an illegal schema, and correcting them.Schema violations would occur during the import-ldif step.

    a. Run `ds2oud -migrateUserSchema`

       ```
       /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --migrateUserSchema --
       odseeBindDN "cn=Directory Manager" --odseeHostname <HOSTNAME> --odseePort 1393 --
       odseeBindPasswordFile /tmp/passwd --oudBindDN "cn=Directory Manager" --
       oudHostname <HOSTNAME> --oudPort 1389 --oudBindPasswordFile /tmp/passwd --
       oudAdminPort 1444 --no-prompt
       ```

       Note:

       This might take all the ODSEE user schema into OUD, including obsolete schema.

    b. **Note about extra schema files in config/schema**

       Note that there were no schema files in config/schema prior to running the command, and after there is only **99-user.ldif**. Running DSsetup later pulls in some other files into the config/schema area due to overwriting of OUD default schema. The various files are: **00-core.ldif**, **05-solaris.ldif** , and **05-oraclefa.ldif**. The middlename is in **05-oraclefa.ldif**, the location is in **00-core.ldif**, the mail rfc822mailbox is in **00-core.ldif**, and the mgrpRFC822MailMember is in **05-solaris.ldif**.

       In a pristine (fresh) OUD instance:

       ```
       attributeTypes: ( 2.16.840.1.113894.200.1.3 NAME 'middleName' SUP name SINGLE-
       VALUE USAGE userApplications )
       attributeTypes: ( 1.3.6.1.4.1.26027.2.1.71 NAME 'location' SYNTAX
       1.3.6.1.4.1.26027.2.5.2 SINGLE-VALUE X-ORIGIN 'OUD' )
       attributeTypes: ( 0.9.2342.19200300.100.1.3 NAME ( 'mail' 'rfc822Mailbox' )
       EQUALITY caseIgnoreIA5Match SUBSTR caseIgnoreIA5SubstringsMatch
       SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} X-ORIGIN 'RFC 4524' )

       attributeTypes: ( 2.16.840.1.113730.3.1.30 NAME 'mgrpRFC822MailMember' SYNTAX
       1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'Solaris Specific' )
       attributeTypes: ( 2.5.4.41 NAME 'name' EQUALITY caseIgnoreMatch SUBSTR
       caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 {32768} X-ORIGIN
       'RFC 4519' )
       ```

       It is recommended to use the OUD default schema for such items.

> **Note:**
>
> The change for the middle name, location, and mgrpRFC822MailMember. The location and mgrpRFC822Mailmmember are identical to UCS definitions. The middle name is slightly different but will go with the OUD default. However, for mail, you need to use the UCS definition since it defines its syntax to be UTF-8 for EAI (Email Address Internationalization) reasons. So 00-core.ldif appears in config/schema along with 99-user.ldif only once that is done.

> **Note:**
>
> OUD is strict with syntax checking, ensuring adherence to specific rules:
>
> - Attribute values must match their intended formats. If an attribute requires a Distinguished Name (DN), it must be a valid DN, not just any string. For example, for the owner attribute, an input like "testuser@testdomain.com" is rejected due to the "@" symbol, and users should provide the correct DN.
> - Value of an integer attribute is an integer and not a non-numeral character.
> - LDAP entries follow strict guidelines for structural object classes. An entry cannot contain two structural object classes. Examples include "groupofuniquenames" and "groupofurls" in a mailing list or "inetResource" and "account" in the same entry.

12. **Process migrateConfiguration**

    This will generate a script to migrate the ODSEE configuration to OUD using "dsconf".

    a. Run `ds2oud -migrateConfiguration`

    ```
    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --migrateConfiguration --
    odseeBindDN "cn=Directory Manager" --odseeHostname <HOSTNAME> --odseePort 1393 --
    odseeBindPasswordFile /tmp/passwd --oudBindDN "cn=Directory Manager" --
    oudHostname <HOSTNAME> --oudPort 1389 --oudBindPasswordFile /tmp/passwd --
    oudAdminPort 1444 --batchFile /tmp/migrate-config --no-prompt
    ```

    > **Note:**
    >
    > - If you run without –no-prompt
    >
    >   DELETE if it asks to create index displayName on piserverdb - say NO
    >
    >   DELETE if it asks to create index cosspecifier on usergroup - say NO
    >
    >   DELETE if it asks to create index inetDomainBaseDN on usergroup - say NO
    > - For schema 1, **inetDomainBaseDN** would be on the DC tree "internet" instead of "usergroup"

    b. Edit /tmp/migrate-config

    With –no-prompt, edit the migrate-config file and remove the following cases:

i. displayName

ii. cosspecifier

iii. inetDomainBaseDN

iv. icsCalendarOwned

Sample Session

```
# /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --migrateConfiguration --
odseeBindDN 'cn=Directory Manager' --odseeHostname <HOSTNAME> --odseePort 1393 --
odseeBindPasswordFile /tmp/passwd --oudBindDN'cn=Directory Manager' --
oudHostname <HOSTNAME> --oudPort 1389 --oudBindPasswordFile /tmp/passwd --
oudAdminPort 1444 --batchFile /tmp/migrate-config --no-prompt
** Naming context(s) available on the ODSEE server :
o=comms-config
o=mlusers
o=pab
o=PiServerDb
o=usergroup
Creation of naming context o=comms-config
Creation of naming context o=mlusers
Creation of naming context o=pab
Creation of naming context o=PiServerDb
Creation of naming context o=usergroup
** Global Configuration Parameters
Configuration of the Global Parameters
** Global ACIs
No action was required, the default OUD configuration applies
** Indexes
<...output snipped...>
** Default Build-in Plugins
** Default Password Policy
Configuration of the Default Password Policy
```

Sample Session

Edited **/tmp/migrate-config** to fix the following:

```
sed -e /inetDomainBaseDN/d -e /cosspecifier/d -e /displayname/d -e /
icsCalendarOwned/d /tmp/migrate-config > /tmp/migrate-config.new
```

After all such replacements that are required, move/rename this **migrate-config.new** as latest **/tmp/migrate-config** for operations in the next steps.

13. **Process migrate-config**

Run **dsconf -F migrate-config**

```
/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsconfig -F /tmp/migrate-config -n -X
-p 1444 -D "cn=Directory Manager" -j /tmp/passwd
```

> **Note:**
>
> This command creates naming contexts and indexes.

Sample Session

```
# /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsconfig -F /tmp/migrate-config -n -
X -p 1444 -D 'cn=Directory Manager' -j /tmp/passwd
```

```
create-workflow-element --set base-dn:o=comms-config --set enabled:true --type db-
local-backend --element-name comms-config -n create-workflow --set base-dn:o=comms-
config --set enabled:true --set workflow-element:comms-config --workflow-name comms-
config_workflow -n set-network-group-prop --group-name network-group --add
workflow:comms-config_workflow -n

create-workflow-element --set base-dn:o=mlusers --set enabled:true --type db-local-
backend --element-name mlusers -n create-workflow --set base-dn:o=mlusers --set
enabled:true --set workflow-element:mlusers --workflow-name mlusers_workflow -n set-
network-group-prop --group-name network-group --add workflow:mlusers_workflow -n

create-workflow-element --set base-dn:o=pab --set enabled:true --type db-local-
backend --element-name pab -n create-workflow --set base-dn:o=pab --set enabled:true
--set workflow-element:pab --workflow-name pab_workflow -n set-network-group-prop --
group-name network-group --add workflow:pab_workflow -n

create-workflow-element --set base-dn:o=PiServerDb --set enabled:true --type db-
local-backend --element-name PiServerDb -n create-workflow --set base-
dn:o=PiServerDb --set enabled:true --set workflow-element:PiServerDb --workflow-name
PiServerDb_workflow -n set-network-group-prop --group-name network-group --add
workflow:PiServerDb_workflow -n

create-workflow-element --set base-dn:o=usergroup --set enabled:true --type db-local-
backend --element-name usergroup -n create-workflow --set base-dn:o=usergroup --set
enabled:true --set workflow-element:usergroup --workflow-name usergroup_workflow -n
set-network-group-prop --group-name network-group --add workflow:usergroup_workflow -
n

 <...output snipped...>
```

14. **List OUD backends**

    Run tests

    - `/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/list-backends`
    - `/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/status -sn`
    - **ldapsearch:**

      ```
      /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ldapsearch -T -X -h HOSTNAME -p
      1389 -D 'cn=Directory Manager' -j /tmp/passwd -b 'o=usergroup' -s sub
      '(objectclass=*)'
      ```

    Sample Search

    ```
    # /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/list-backends

    Backend ID : Base DN

    ---------------:------------------

     PiServerDb : o=PiServerDb

     adminRoot : cn=admin data

     ads-truststore : cn=ads-truststore

     backup : cn=backups

     comms-config : o=comms-config

     mlusers : o=mlusers

     monitor : cn=monitor
    ```

```
 pab : o=pab

 schema : cn=schema

 tasks : cn=tasks

 usergroup : o=usergroup

 virtualAcis : cn=virtual acis

 # /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/status -sn

Server Run Status: Started

Open Connections: 0

Host Name: <HOSTNAME>

Administrative Users: cn=Directory Manager

Installation Path: /opt/oracle/Oracle/Middleware/oud

Instance Path: /opt/oracle/Oracle/Middleware/asinst_1/OUD

Version: Oracle Unified Directory 12.2.1.4.0

<...output snipped...>
```

15. **Enable replication on ODSEE**

Enable ODSEE replication, but do not create a replication agreement syntax: `dsconf enable-repl -h host -p port -d ReplicaID master suffix-DN`. Run the command for each suffix.

- `/opt/dsee7/bin/dsconf enable-repl -p 1393 -pwd-file /tmp/passwd -d 1 master o=usergroup`
- `/opt/dsee7/bin/dsconf enable-repl -p 1393 -pwd-file /tmp/passwd -d 1 master o=mlusers`
- `/opt/dsee7/bin/dsconf enable-repl -p 1393 -pwd-file /tmp/passwd -d 1 master o=PiServerDb`
- **For schema 1 only:** `/opt/dsee7/bin/dsconf enable-repl -p 1393 -pwd-file /tmp/passwd -d 1 master o=internet`

Sample Session

```
# /opt/dsee7/bin/dsconf enable-repl -p 1393 --pwd-file /tmp/passwd -d 1 master
o=usergroup

Use "dsconf create-repl-agmt" to create replication agreements on "o=usergroup".

# /opt/dsee7/bin/dsconf enable-repl -p 1393 --pwd-file /tmp/passwd -d 1 master
o=mlusers

Use "dsconf create-repl-agmt" to create replication agreements on "o=mlusers".

# /opt/dsee7/bin/dsconf enable-repl -p 1393 --pwd-file /tmp/passwd -d 1 master
o=PiServerDb

Use "dsconf create-repl-agmt" to create replication agreements on "o=PiServerDb".
```

16. **Setup the OUD replication gateway**

    Run `oud-replication-gateway-setup`.

    > **Note:**
    >
    > For schema 1 add: `–baseDN o=internet`

```
/opt/oracle/Oracle/Middleware/Oracle_OUD1/oud-replication-gateway-setup --cli --
hostname <HOSTNAME> --adminConnectorPort 1445 --replicationPortForLegacy 1390 --
rootUserDN "cn=Directory Manager" --rootUserPasswordFile /tmp/passwd --baseDN
o=usergroup --baseDN o=mlusers --baseDN o=PiServerDb --hostNameLegacy <HOSTNAME> --
portLegacy 1393 --doNotUpdateTrustStoreWithLegacyCertsArg --bindDNLegacy
"cn=Directory Manager" --bindPasswordFileLegacy /tmp/passwd --hostNameNg <HOSTNAME>
--portNg 1444 --adminUID admin --adminPasswordFile /tmp/passwd --trustAll --no-
prompt --noPropertiesFile --doNotMonitorUsingDsccLegacy --replicationPortNg 1989 --
verbose --bindDNNg 'cn=Directory Manager' --bindPasswordFileNg /tmp/passwd
```

Sample Session

```
#/opt/oracle/Oracle/Middleware/oud/oud-replication-gateway-setup --cli --hostname
localhost --adminConnectorPort 2445 --replicationPortForLegacy 1391 --rootUserDN
"cn=Directory Manager" --rootUserPasswordFile /tmp/passwd --baseDN o=usergroup --
baseDN o=mlusers --baseDN o=PiServerDb --hostNameLegacy localhost --portLegacy 1389
--doNotUpdateTrustStoreWithLegacyCertsArg --bindDNLegacy "cn=Directory Manager" --
bindPasswordFileLegacy /tmp/passwd --hostNameNg localhost --portNg 1444 --adminUID
admin --adminPasswordFile /tmp/passwd --trustAll --noPropertiesFile --
doNotMonitorUsingDsccLegacy --replicationPortNg 1989 --verbose --bindDNNg
'cn=Directory Manager' --bindPasswordFileNg /tmp/passwd

Oracle Unified Directory 12.2.1.4.0

Please wait while the replication gateway setup program initializes ..... Done.

Once the setup of the replication gateway will be completed (if not already done)
you have to initialize the contents of the Oracle Unified Directory servers with the
contents of the ODSEE server for replication to work.

You can follow these steps to synchronize the contents of the replicated base DNs:

1. Run the following command in the ODSEE host (<HOSTNAME>):

dsadm export \

-f opends-export \

/var/opt/sun/directory/ds7 \

o=usergroup \

o=mlusers \

o=PiServerDb \

{exportedLDIFPath}

Where {exportedLDIFPath} is the path of the resulting LDIF file containing the
replicated data.
```

2. Run the following command:

```
<instancePath>/bin/dsreplication pre-external-initialization \

--hostname <HOSTNAME> \

--port 1444 \

--adminUID admin \

--adminPasswordFile ****** \

--baseDN o=usergroup \

--baseDN o=mlusers \

--baseDN o=PiServerDb \

--trustAll \

--no-prompt \

--noPropertiesFile
```

3. Copy the LDIF file generated in the first step in a directory accessible by the Oracle Unified Directory servers and run the following command for every Oracle Unified Directory server that contains data to be replicated:

```
<instancePath>/bin/import-ldif \

--hostname <HOSTNAME> \

--port 1444 \

--bindDN cn=Directory\ Manager \

--bindPasswordFile ****** \

--includeBranch o=usergroup \

--includeBranch o=mlusers \

--includeBranch o=PiServerDb \

--ldifFile {exportedLDIFPath} \

--clearBackend \

--trustAll \

--noPropertiesFile
```

4. Run the following command:

```
<instancePath>/bin/dsreplication post-external-initialization \

--hostname <HOSTNAME> \

--port 1444 \

--adminUID admin \
--adminPasswordFile ****** \

--baseDN o=usergroup \

--baseDN o=mlusers \

--baseDN o=PiServerDb \

--trustAll \

--no-prompt \

--noPropertiesFile

<...output snipped...>

The replication gateway setup has completed successfully
```

17. **Global admin is created**

    The "global admin" is created when you run `oud-replication-gateway-setup`. You can verify that by doing a `ldapsearch` for `cn=admin`, , and `cn=admin data`.

    Run `ldapsearch`.

```
/opt/oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -X -h <HOSTNAME> -p 1444
-D 'cn=Directory Manager' -j /tmp/passwd --useSSL -b 'cn=Administrators,cn=admin
data' -s sub '(objectclass=*)'
```

Sample Session

```
 # /opt/oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -X -h HOSTNAME -p
1444 -D 'cn=Directory Manager' -j /tmp/passwd --useSSL -b
'cn=Administrators,cn=admin data' -s sub '(objectclass=*)'
'cn=Administrators,cn=admin data' -s sub '(objectclass=*)'

 dn: cn=Administrators,cn=admin data

objectClass: top

objectClass: groupofurls

description: Group of identities which have full access.

cn: Administrators

memberURL: ldap:///cn=Administrators,cn=admin data??one?(objectclass=*)



dn: cn=admin,cn=Administrators,cn=admin data

userPassword: {SSHA512}YvhmnmRBgN8sAQHFffwTTd4XR0JT+U2GtN4kx3L9a6uBO68uKpqGiifL\

/kV3XdyzaUjjcJsPts9DA6mPaRj55URa5aHkaGTX

objectClass: person

objectClass: top

description: The Administrator that can manage all the server instances.

 <...output snipped...>
```

18. **Process DSsetup 6.4.0.30.0 to pull in the corrected schema**

    This is the second run of DSsetup 6.4.0.30.0 (the first time was in Step 16 above).

    > **Note:**
    >
    > You must run DSsetup 6.4.0.30.0 at least once before import-ldif, otherwise entries are not pulled in due to schema violations. It is important that you match the schema type and u/g suffix that exists on the ODSEE side

    .

    > **Note:**
    >
    > For schema 1: specify `-schemaType 1 -dctree o=internet`

    ```
    rundssetup
    bin/rundssetup --dsType=OUD\

     --instanceLocation /opt/oracle/Oracle/Middleware/asinst_1 \

     --bindPasswordFile /tmp/passwd \

     --schemaType 2 \

     --addIndex no \

     --reIndex no \

     --ugtree o=usergroup \

     --updateSchema yes \

     --modifyDS yes
    ```

19. **Export ODSEE data to ldif Again and Cleanup/prepare**

    You must do this export again after **oud-replication-gateway-setup** was run (as it is known to update ODSEE). Hence, you must use this exported ODSEE data after you run **oud-replication-gateway-setup**. Recheck using **ds2oud** and fix any invalid entries. Ensure this ldif file is validated successfully against OUD Schema and is ready for import in the next steps.

    a. **Run dsconf export**

       > **Note:**
       >
       > For schema 1 add: `o=internet`. Also use "output-not-folder" option when running this export command, so that data is exported without folding/ truncation (it enables correct search/ replace in the next steps).

```
/opt/dsee7/bin/dsconf export --accept-cert --user-dn 'cn=Directory Manager' --
pwd-file /tmp/passwd -f opends-export -f output-not-folded -h <HOSTNAME> -p 1393
o=usergroup o=mlusers o=PiServerDb odsee-data2.ldif
```

b. **Copy it to /tmp , Cleanup and Check ds2oud**

```
cp /var/opt/sun/directory/ds7/logs/odsee-data2.ldif /tmp/
odsee_before_roledn_rep.ldif

 Fixed any occurrences of "roledn" or "groupdnattr" :

 sed "s@) roledn@) groupdn@;s@) groupdnattr@) groupdn@;s@)roledn@)groupdn@;s@and
roledn@and groupdn@;s@or

 roledn@or groupdn@" /tmp/odsee_before_roledn_rep.ldif > /tmp/odsee-data2.ldif

 Ran ds2oud :

 /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/ds2oud --ldifDBFile /tmp/odsee-
data2.ldif --userSchemaFile /opt/sun/comms/dssetup/lib/foranalysis-oud-
schema.ldif
```

> **Note:**
>
> Diagnose ODSEE LDIF data file: **/tmp/odsee-data2.ldif**

The data was validated successfully regarding the OUD schema. This file **/tmp/odsee-data2.ldiff** is now ready for import into OUD.

20. Run dsreplication pre-external-initialization

for schema 1: add -baseDN o=internet.

```
/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsreplication pre-external-
initialization --hostname <HOSTNAME> --port 1444 --adminUID admin --
adminPasswordFile /tmp/passwd --baseDN o=usergroup --baseDN o=mlusers --baseDN
o=PiServerDb --trustAll --no-prompt --noPropertiesFile
```

Sample Session

```
# /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsreplication pre-external-
initialization --hostname <HOSTNAME> --port 1444 --adminUID admin --
adminPasswordFile /tmp/passwd --baseDN o=usergroup --baseDN o=mlusers --baseDN
o=PiServerDb --trustAll --no-prompt --noPropertiesFile

Establishing connections ..... Done.

Preparing base DN o=mlusers to be initialized externally ..... Done.

Preparing base DN o=PiServerDb to be initialized externally ..... Done.

Preparing base DN o=usergroup to be initialized externally ..... Done.

Now you can proceed to the initialization of the contents of the base DN's on all
the replicated servers. You can use the command import-ldif or the binary copy to do
so. You must use the same LDIF file or binary copy on each server.

 When the initialization is completed you must use the subcommand 'post-external-
initialization' for replication to work with the new base DN's contents.
```

```
 See /var/tmp/oud-replication-3459260775445051714.log for a detailed log of this
operation.
```

21. Run `import-ldif` into OUD

    The ODSEE data prepared above **/tmp/odsee-data2.ldiff** is now imported into OUD, using the respective backend IDs.

    a. Run list-backends to find out Backend ID to use for **import-ldif**

    ```
    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/list-backends
    ```

    ### Sample Session

    ```
    # /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/list-backends
    Backend ID : Base DN
    ---------------:------------------
    PiServerDb : o=PiServerDb
    adminRoot : cn=admin data
    ads-truststore : cn=ads-truststore
    backup : cn=backups
    comms-config : o=comms-config
    mlusers : o=mlusers
    monitor : cn=monitor
    pab : o=pab
    schema : cn=schema
    tasks : cn=tasks
    usergroup : o=usergroup
    virtualAcis : cn=virtual acis
    ```

    b. Run **import-ldif**

    > **Note:**
    >
    > Use backendID obtained from list-backends above.

    ```
    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/import-ldif --hostname <HOSTNAME>
    --port 1444 --bindDN cn=Directory\ Manager --bindPasswordFile /tmp/passwd --
    includeBranch o=usergroup --backendID usergroup --ldifFile /tmp/odsee-data2.ldif
    --clearBackend --trustAll --noPropertiesFile

    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/import-ldif --hostname <HOSTNAME>
    --port 1444 --bindDN cn=Directory\ Manager --bindPasswordFile /tmp/passwd --
    includeBranch o=mlusers --backendID mlusers --ldifFile /tmp/odsee-data2.ldif --
    clearBackend --trustAll --noPropertiesFile

    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/import-ldif --hostname <HOSTNAME>
    --port 1444 --bindDN cn=Directory\ Manager --bindPasswordFile /tmp/passwd --
    includeBranch o=PiServerDb --backendID PiServerDb --ldifFile /tmp/odsee-
    data2.ldif --clearBackend --trustAll --noPropertiesFile
    ```

    For schema 1 only:

    ```
    /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/import-ldif --hostname <HOSTNAME>
    --port 1444 --bindDN cn=Directory\ Manager --bindPasswordFile /tmp/passwd --
    includeBranch o=internet --backendID internet --ldifFile /tmp/odsee-data2.ldif --
    clearBackend --trustAll --noPropertiesFile
    ```

22. Run `dsreplication post-external-initialization`

    ```
    dsreplication post-external-initialization
    ```

> **Note:**
>
> For schema 1 add:
>
> ```
>  -baseDN o=internet
> ```

```
/opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsreplication post-external-
initialization --hostname <HOSTNAME> --port 1444 --adminUID admin --
adminPasswordFile /tmp/passwd --baseDN o=usergroup --baseDN o=mlusers --baseDN
o=PiServerDb --trustAll --no-prompt --noPropertiesFile
```

Sample Session:

```
# /opt/oracle/Oracle/Middleware/asinst_1/OUD/bin/dsreplication post-
external-initialization --hostname <HOSTNAME> --port 1444 --adminUID admin
--adminPasswordFile /tmp/passwd --baseDN o=usergroup --baseDN o=mlusers --
baseDN o=PiServerDb --trustAll --no-prompt --noPropertiesFile

 Establishing connections ..... Done.

 Executing post-external initialization on base DN o=mlusers ..... Done.

 Executing post-external initialization on base DN o=PiServerDb ..... Done.

 Executing post-external initialization on base DN o=usergroup ..... Done.

 Post initialization procedure completed successfully.

 See /var/tmp/oud-replication-3702816444427427726.log for a detailed log
of this operation.
```

23. **Test Replication**

    To verify that replication is working write an attribute to ODSEE and see if it shows up on the OUD side.

    Example shown below is with ldapmodify and ldapsearch commands (used on sample 'testuser1' account):

    Sample Session:

```
# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
<HOSTNAME> -p 1389 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

dataSource: Messaging Server Initial Configuration

mailHost: <HOSTNAME>

objectClass: person

objectClass: ipUser

objectClass: organizationalPerson
```

```
objectClass: inetOrgPerson

objectClass: top

objectClass: userPresenceProfile

objectClass: inetUser

objectClass: inetLocalMailRecipient

objectClass: iplanet-am-managed-person

objectClass: inetMailuser

mailUserStatus: active

inetUserStatus: active

uid: testuser1

cn: testuser1

sn: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

mail: testuser1@example.com

mailDeliveryOption: mailbox


# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
<HOSTNAME> -p 1393 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

objectClass: top

objectClass: person

objectClass: inetOrgPerson

objectClass: organizationalPerson

objectClass: iplanet-am-managed-person

objectClass: inetUser

objectClass: ipUser

objectClass: userPresenceProfile

objectClass: inetMailuser
```

```
objectClass: inetLocalMailRecipient

sn: testuser1

cn: testuser1

uid: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

inetUserStatus: active

mailDeliveryOption: mailbox

dataSource: Messaging Server Initial Configuration

mailUserStatus: active

mail: testuser1@example.com

mailHost: <HOSTNAME>



# cat /tmp/add.ldif

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

changetype: modify

add: mailEquivalentAddress

mailEquivalentAddress: testuser1@example.com



# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapmodify -h <HOSTNAME> -
p 1393 -D 'cn=Directory Manager' -j /tmp/passwd --filename /tmp/add.ldif

Processing MODIFY request for
uid=testuser1,ou=People,o=example.com,o=usergroup

MODIFY operation successful for DN
uid=testuser1,ou=People,o=example.com,o=usergroup



# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
<HOSTNAME> -p 1393 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

mailEquivalentAddress: testuser1@example.com
```

```
objectClass: top

objectClass: person

objectClass: inetOrgPerson

objectClass: organizationalPerson

objectClass: iplanet-am-managed-person

objectClass: inetUser

objectClass: ipUser

objectClass: userPresenceProfile

objectClass: inetMailuser

objectClass: inetLocalMailRecipient

sn: testuser1

cn: testuser1

uid: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

inetUserStatus: active

mailDeliveryOption: mailbox

dataSource: Messaging Server Initial Configuration

mailUserStatus: active

mail: testuser1@example.com

mailHost: <HOSTNAME>



# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
<HOSTNAME> -p 1389 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

 dn: uid=testuser1,ou=People,o=example.com,o=usergroup

dataSource: Messaging Server Initial Configuration

mailHost: <HOSTNAME>

mailEquivalentAddress: testuser1@example.com

objectClass: person
```

```
objectClass: inetOrgPerson

objectClass: organizationalPerson

objectClass: ipUser

objectClass: top

objectClass: inetUser

objectClass: userPresenceProfile

objectClass: iplanet-am-managed-person

objectClass: inetLocalMailRecipient

objectClass: inetMailuser

uid: testuser1

inetUserStatus: active

mailUserStatus: active

cn: testuser1

sn: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

mail: testuser1@example.com

mailDeliveryOption: mailbox


# cat /tmp/add.ldif

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

changetype: modify

add: mailEquivalentAddress

mailEquivalentAddress: testuser1alt@example.com


# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapmodify -h <HOSTNAME> -
p 1389 -D 'cn=Directory Manager' -j /tmp/passwd --filename /tmp/add.ldif

Processing MODIFY request for
uid=testuser1,ou=People,o=example.com,o=usergroup

MODIFY operation successful for DN
uid=testuser1,ou=People,o=example.com,o=usergroup
```

```
# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
<HOSTNAME> -p 1389 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

dn: uid=testuser1,ou=People,o=example.com,o=usergroup

dataSource: Messaging Server Initial Configuration

mailEquivalentAddress: testuser1@example.com

mailEquivalentAddress: testuser1alt@example.com

mailHost: <HOSTNAME>

objectClass: person

objectClass: ipUser

objectClass: organizationalPerson

objectClass: inetOrgPerson

objectClass: top

objectClass: userPresenceProfile

objectClass: inetUser

objectClass: inetLocalMailRecipient

objectClass: iplanet-am-managed-person

objectClass: inetMailuser

mailUserStatus: active

inetUserStatus: active

uid: testuser1

cn: testuser1

sn: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

mail: testuser1@example.com

mailDeliveryOption: mailbox


# /opt/oracle/Oracle/Middleware/Oracle_OUD1/bin/ldapsearch -T -h
```

```
<HOSTNAME> -p 1393 -D 'cn=Directory Manager' -j /tmp/passwd -b
'o=usergroup' -s sub '(uid=testuser1)'

 dn: uid=testuser1,ou=People,o=example.com,o=usergroup

mailEquivalentAddress: testuser1@example.com

mailEquivalentAddress: testuser1alt@example.com

objectClass: top

objectClass: person

objectClass: inetOrgPerson

objectClass: organizationalPerson

objectClass: iplanet-am-managed-person

objectClass: inetUser

objectClass: ipUser

objectClass: userPresenceProfile

objectClass: inetMailuser

objectClass: inetLocalMailRecipient

sn: testuser1

cn: testuser1

uid: testuser1

userPassword: {SSHA}g02arnhXqR7S7Qc10Z9MhGnvh+cpdzwY4FfOGA==

inetUserStatus: active

mailDeliveryOption: mailbox

dataSource: Messaging Server Initial Configuration

mailUserStatus: active

mail: testuser1@example.com

mailHost: <HOSTNAME>
```

24. **Switch UCS products from ODSEE to OUD**

    Refer to product specific documentation to switch the UCS products in your deployment to this OUD as the directory service backend.

    For each product, refer to its LDAP configuration-related parameter names, to ensure all relevant LDAP settings are now switched to this OUD Hostname (FQDN) and ports. It must be done on all your product instances (based on single or distributed deployment).

Example: For MS Product, this hostname and port can be set using configuration parameters: local.ugldaphost and local.ugldapport. Similarly, each UCS product has its own configuration parameters for LDAP settings and it must be set now to OUD.

> **Note:**
>
> If you see any issues with OCUCS Admin Password Policy (example: in cases like Calendar or Contact Servers), then you will have to re-run that product-specific configurator with this backend OUD instance setup.

Setting up Loosely Coupled Migration

To use a loosely coupled migration scenario instead of a tightly coupled migration scenario, you may add a switch to the oud-replication-gateway-setup "**–doNotSendUpdateToLegacyServer**"

Setting up a direct transition migration

Instead of a tightly coupled migration scenario, to do one-off direct transition/migration, follow the procedure of exporting from ODSEE, and importing that data into OUD. You will have to ensure the following :

- ODSEE data exported into ldif
- Prepare ODSEE data ldif : Diagnose using ds2oud, Fix/clean up any invalid or incompatible issues that is flagged. Ensure this ODSEE ldif file is validated successfully against OUD's schema
- Import that ODSEE ldif into OUD.

Uninstall Commands

This topic outlines the process of uninstalling OUD, ODSEE, and replication gateway instances.

Uninstall Replication Gateway Instance

```
/opt/oracle/Oracle/Middleware/asinst_2/OUD/uninstall \
--cli \
--hostname <HOSTNAME> \
--adminUID admin \
--adminPasswordFile /tmp/passwd \
--bindDNLegacy cn=Directory\ Manager \
--bindPasswordFileLegacy /tmp/passwd \
--trustAll \
--no-prompt \
--noPropertiesFile
```

Uninstall OUD Instance

```
/opt/oracle/Oracle/Middleware/asinst_1/OUD/uninstall \
--cli \
--remove-all \
-h <HOSTNAME> \
--adminUID admin \
-j /tmp/passwd \
--trustAll \
--no-prompt \
--noPropertiesFile
```

Uninstall ODSEE

```
/opt/dsee7/bin/dsadm delete/var/opt/sun/directory/ds7
```