# Oracle® Communications Calendar Server

Installation and Configuration Guide

Release 7.0.5

**E54934-01**

February 2015

ORACLE®

Oracle Communications Calendar Server Installation and Configuration Guide, Release 7.0.5

E54934-01

# Contents

## 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



## 8 Calendar Server Post-Installation Tasks



## 9 Upgrading Calendar Server

## 10 Creating a Calendar Server 6 and 7 Coexistent Deployment



## 11 Migrating to Calendar Server

## 12 Uninstalling Calendar Server



## 13 Installing Patches



## A Calendar Server Configuration Scripts

# 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:

- Oracle Communications Unified Communications Suite component products
- MySQL Server or Oracle Database
- Oracle GlassFish Server
- Oracle Directory Server Enterprise Edition and LDAP
- System administration and networking

## Related Documents

For more information, see the following documents in the Calendar Server documentation set:

- *Calendar Server Concepts*: Provides information on the concepts associated with Calendar Server.
- *Calendar Server System Administrator's Guide*: Provides instructions for administering Calendar Server.
- *Calendar Server Release Notes*: Describes the fixes, known issues, troubleshooting tips, and required third-party products and licensing.
- *Calendar Server Security Guide*: Provides guidelines and recommendations for setting up Calendar Server in a secure configuration.
- *Calendar Server WCAP Developer's Guide*: Describes how to use the Web Calendar Access Protocol 7.0 (WCAPbis) with Calendar Server.

## 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.

# 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
- Oracle GlassFish Server
- Either MySQL Server or Oracle Database
- Calendar Server

Calendar Server depends on Oracle Directory Server Enterprise Edition for LDAP services. If your site does not currently have Directory Server deployed and you need to install it, see the Oracle Directory Server Enterprise Edition documentation for instructions, at:

http://docs.oracle.com/cd/E29127_01/index.htm

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
   - Oracle GlassFish Server
   - Either MySQL Server or Oracle Database
4. Prepare the Directory Server schema by installing and running the most current **comm_dssetup** script from the Communications Suite distribution.
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 by running the Unified Communications Suite installer in either interactive or silent mode. When you run the Communications Suite 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.

For more information, see the topic on installing Unified Communications Suite in silent mode in *Unified Communications Suite Installation Guide* at:

https://wikis.oracle.com/display/CommSuite

## Ensuring a Successful Calendar Server Installation

Only qualified personnel should install the product. You must be familiar with the UNIX operating system and Oracle GlassFish 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 GlassFish 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**. |

# 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 (also referred to as Calendar Server 7 and formerly known as Oracle Communications Calendar Server for CALDAV Clients and Sun Java System 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

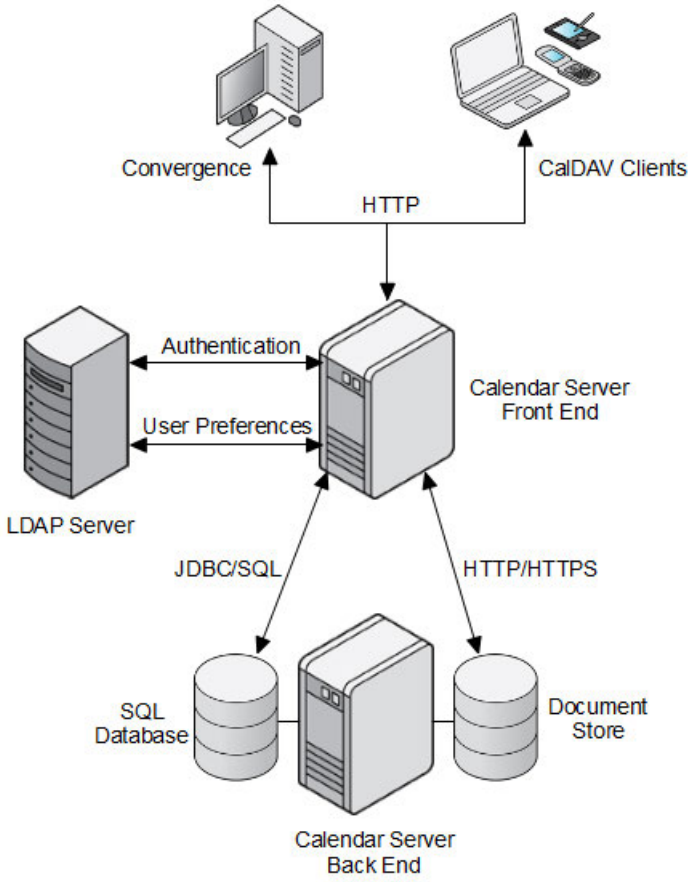
Calendar Server consists of the following front-end and back-end components:

- Stateless J2EE-based front end (provided by Oracle GlassFish Server)
- SQL back end (can be either MySQL Server or Oracle Database)
- 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 SQL database and the document store. The document store can be located on a separate host if desired.

*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, either by using a simple load balancer or by using the GlassFish Server cluster functionality.

## 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 GlassFish Server's web container hosting the server is configured with corresponding JNDI resources containing the actual back-end information (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

Calendar Server requires that you use GlassFish Server as its web container. 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.

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 contact 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 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 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 the iSchedule Database

The iSchedule database is used to manage external calendar invitations. You must set up one iSchedule database per configured Calendar Server back-end database. That is, every logically configured database must have its own unique iSchedule database.

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 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
- Using Load Balancing
- Planning Backup Strategies

### Planning for High Availability

You can configure Calendar Server to be highly available. Choices for making Calendar Server highly available include GlassFish Server high availability features, 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 persistent connections) is necessary to distribute the load across the front-end hosts. Multiple front-end hosts can be grouped together, either by using a simple load balancer or by using the GlassFish Server cluster functionality.

### 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

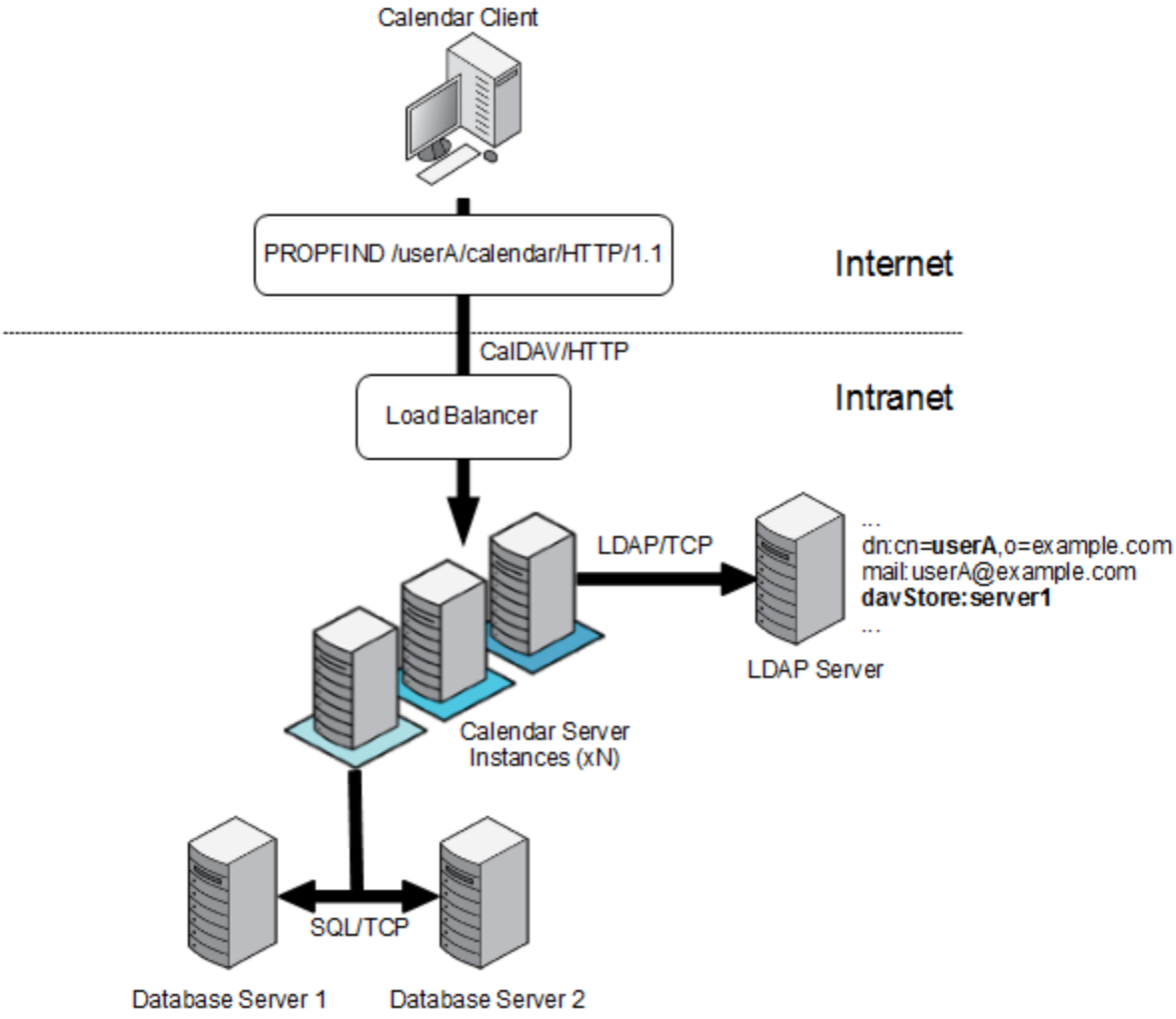
When planning your Calendar Server logical architecture, you can use the following options:

- **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*.

# 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 an Oracle GlassFish Server 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 10 and 11 | SPARC, x64 | See the Oracle Solaris documentation for patch information. |
| Oracle Linux 6 and Red Hat Enterprise Linux 6 64-bit | x64 | See the Oracle Linux and Red Hat Enterprise 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<br>To upgrade to Oracle Database 12c from Oracle Database 11g, see *Oracle Database Upgrade Guide* at:<br>https://docs.oracle.com/database/121/UPGRD/toc.htm |
| MySQL Server | 5.5.32, 5.6<br>To upgrade to MySQL Server 5.6 from MySQL Server 5.5, see "Upgrading from MySQL 5.5 to 5.6" at:<br>http://dev.mysql.com/doc/refman/5.6/en/upgrading-from-previous-series.html |

**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 |
|---|---|---|
| Oracle Directory Server Enterprise Edition | 6.x, 7, 11gR1 Patch Set 2 (11.1.1.7.0) | If doing a fresh installation, use 11gR1. |
| Oracle GlassFish Server | 3.1.2.x | Required as the web container. Download the patch from My Oracle Support at: `https://support.oracle.com` |
| Java | 7 | Be sure to install the latest security update. |

# 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 | None. |
| Connector for Microsoft Outlook | 8 | None. |
| 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 cleitn | 6.0.1 | Support for scheduling in iOS 4.x on iPhone, iPod touch, and iPad (iOS 7 is the latest version). |
| Lightning | 1.9.1 | Lightning 1.9.1 is the latest version. |
| eM | Not applicable | Desktop client for Window (.NET) that supports CalDAV access. This client has not officially been tested by Oracle. In general, any CalDAV compliant client should work. |
| Enterprise Tasks | Not applicable | eTask is a business applications for the iPhone and iPod touch that uses the CalDAV protocol to view, edit, create, delete, and sync your tasks on the CalDAV server. |

*Table 3–4 (Cont.) Calendar Server Clients*

| Client | Minimum Version | Notes |
|---|---|---|
| Android CalDAV-Sync | Not applicable | CalDAV-Sync is a CalDav client for Android to synchronize events. Testing done with CalDAV sync client version 0.3.3 and Task Client version 1.0.3. |
| NotifyLink | 4.7 | Depending on the device, might require version 4.7 or 4.8. |

# 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 installed on a single managed server in a GlassFish Server domain.

*Table 3–5 Minimum Hardware Requirements*

| Component | Requirement |
|---|---|
| Disk Space | Approximately 20 MB required for Calendar Server software. |
| 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 | Comments |
|---|---|---|
| Directory to store configuration and data files | **/var/opt/sun/comms/davserver** | No comments. |
| Runtime user ID under which Calendar Server runs | **root** | This user must match the user that runs the Glassfish Server instance. It is also the owner of the application data and configuration directory. |
| Runtime group to contain Calendar Server runtime user ID | **bin** | This group must match the group of the user that runs the Glassfish Server instance. It is also the owner of the application data and configuration directory. |
| Fully qualified host name of this system | FQDN of host | No comments. |

## 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** |
| 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

Table 3–10 lists the document store information that you provide during initial configuration.

*Table 3–10 Document Store Information*

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

## iSchedule Database Information

Table 3–11 lists the iSchedule database information that you provide during initial configuration.

*Table 3–11 iSchedule Database Information*

| Information Type | Default Value |
|---|---|
| The iSchedule database 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 iSchedule database only) The name of the host where the remote iSchedule database is located. | No default value. |
| (Remote iSchedule database only) The port number for the remote iSchedule database 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** |
| 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. |

*Table 3–12 (Cont.) GlassFish Server Information*

| Information Type | Default Value |
|---|---|
| URI path of the deployed server | **https://***FQDN of host***:443/** (root directory)<br>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 setting up CalDAV autodiscovery in *Calendar Server System Administrator's Guide*. |

## LDAP Information

Table 3–13 lists the LDAP information that you provide during initial configuration.

*Table 3–13 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 3–14 lists the email notification information that you provide during initial configuration.

*Table 3–14 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 GlassFish Server
- Installing Directory Server
- Installing the Database

## Installing Java

Oracle GlassFish Server is a Java application and needs a Java environment in which to run.

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

To get the Java software, go to the Java SE Downloads at:

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

For information about installing Java, see the *Installing Java for Communications Suite* documentation at:

https://wikis.oracle.com/display/CommSuite

## Installing 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 Directory Server

Calendar Server uses Oracle Directory Server Enterprise Edition 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 it, see the Oracle Directory Server Enterprise Edition documentation at:

http://docs.oracle.com/cd/E29127_01/index.htm

Prior to installing and configuring Calendar Server, you must also prepare the Directory Server LDAP schema by running the **comm_dssetup.pl** script. This script, which is provided as part of the Unified Communications Suite installer, 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* at:

https://wikis.oracle.com/display/CommSuite/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-solaris10-sparc-64bit.pkg
     ```

   - On Linux 6:

     ```
     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:

   http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.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:

     http://www.oracle.com/pls/db112/portal.portal_db?selected=11

   - 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:

     http://docs.oracle.com/database/121/ADMQS/install.htm#ADMQS0232

   - 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 execution 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 Oracle GlassFish 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 media pack for Oracle Communications Unified Communications Suite, which contains the Calendar Server software, from the Oracle software delivery website, located at:

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

2. Copy the Communications Suite distribution ZIP file to a temporary directory on your Calendar Server hosts and extract the files.

## Preparing Directory Server

To prepare Directory Server:

1. On your Directory Server hosts, install and run the **comm_dssetup.pl** script. You must use at least version 6.4-25.01.

   For more information, see the Communications Suite Directory Server setup script documentation at:

   https://wikis.oracle.com/display/CommSuite

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

2. 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 about provisioning users and schema, see *Communications Suite Schema Reference* at:

   https://wikis.oracle.com/display/CommSuite/Communications+Suite+Schema+Reference

## Installing the Calendar Server Software

To install the Calendar Server software:

1. Go to the directory where you extracted the Communications Suite distribution files.
2. Run the Communications Suite Installer.

   ```
   ./commpkg install
   ```

   For more information about running the installer, see the **commpkg install** usage documentation at:

   https://wikis.oracle.com/display/CommSuite

3. Select **Calendar Server** and proceed with the installation.
4. If the installer detects older versions of LDAP tools, you receive the following prompts asking if you want to change the versions.

   You must answer **y**.

   ```
   Do you wish to change pkg versions for LDAPCSDK6 [n] : y

   Do you wish to change pkg versions for LDAPCSDK6 Tools [n] : y
   ```

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/data
   rm -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/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.

# 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 **comm_dssetup.pl** script, which is installed by the Communications Suite installer, 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**.

## 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
   ```

## 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 with secure mode.

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

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

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

   See "Calendar Server Configuration Scripts" for more information.

   ```
   ./init-config
   ```

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

4. 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.
5. 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.

6. Enter the 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
7. 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
8. Enter the GlassFish Server Configuration Details.
   - Install Directory
   - Domain Directory
   - Document Root Directory
   - Server Target Name
   - Virtual Server Identifier
   - GlassFish Server administration server host
   - GlassFish 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, potentially leading to misconfigurations. Using root as the URI path replaces the GlassFish Server's main **index.html** file; thus, the GlassFish welcome page is no longer displayed at

`http://host:port`. The following example URIs explain the behavior of this entry:

- **https:/host.example.com:443** - Not allowed, deployment context cannot be blank.
- **https:/host.example.com:443/** - Deploy in root.
- **https:/host.example.com:443/davserver** - Deploy in **davserver**.

If you do deploy Calendar Server by using a root (**/**) URI context, you cannot deploy Convergence in the same instance of Glass Fish without moving the **/iwc_static/** files somewhere. The Convergence static files are deployed under the default root context and if another application is deployed to **/**, GlassFish is not able to find the **/iwc_static/*** URLs, resulting in a 404 error immediately after login. There are potential workarounds for this issue, but at this time 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*.

9. 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 do not need to perform any SSL setup for the initial configuration. However, you could still 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.

10. Enter the Notification Mail Server Configuration Details.
    - Mail Server Host Name
    - Mail Server Port Number
11. 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.
12. Restart GlassFish Server.

## 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. On each front-end host, running the **config-backend** script
6. On each front-end host, enabling connection pool validation
7. On each front-end host, restarting Oracle GlassFish Server

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

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

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, 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
     ```

     Skip to Step 3.
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."

## 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 GlassFish 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 then restart GlassFish 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

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:portNumb
er=3306:networkProtocol=jdbc" caldav1Pool

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

## 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.

## Java Version

The remote document store and the Oracle GlassFish Server host that functions as the web container for Calendar Server must both be using either Java 6 or Java 7. Do not mix Java 6 and Java 7. If the GlassFish Server host and remote document store host are using different Java versions, SSL errors occur. The result is a `getContentStream` failed error in the browser.

## 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. 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
```

You must do this on all back ends.

2. Restart Calendar Server by restarting the GlassFish Server instance.

## 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 GlassFish Server instance.

## 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 GlassFish Server instance.

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

```
CalendarServer_home/sbin/start-as
CalendarServer_home/sbin/stop-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 GlassFish Server instance.
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 GlassFish Server instance.

# 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 at:

https://wikis.oracle.com/display/CommSuite/Using+the+iSchedule+Channel+to+Handle+iMIP+Messages

## 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 Calendar Server 7. 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.

## 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, " Database Schema Version Changes for Calendar Server Beginning with Release 7 Update 2" 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 Oracle GlassFish Server container, 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 GlassFish Server while 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 | Comments |
|---|---|---|
| 7.0.5.17.0 | Automatic upgrade to version 9. | No schema upgrade performance issues. |
| 7.0.4.16.0 | No database changes. | No comments. |
| 7.0.4.15.0 | No database changes. | No comments. |
| 7.0.4.14.0 | Automatic Upgrade to Version 8.<br>(MySQL Server only) The charset and collation of most tables and fields is changed to **utf8mb4**.<br>■ 8 **ALTER TABLE** commands are run.<br>Column added to CalProp.<br>■ 1 **ALTER TABLE** command is run. | Fairly intensive for MySQL Server. |
| 7.0.3.8.0 | Automatic Upgrade to Version 7.<br>Full table scan fixes missing values in **Resources.resourcetype = 'CAL_RESOURCE';**. | No comments. |
| 7.0.2.4.0 | Automatic Upgrade to Version 6.<br>■ Rename table **Resource** to **Resources**.<br>■ **ALTER TABLE** command run on **CalProp** to change **supported_calendar_component_set** to **supported_cal_component_set**.<br>■ Change table **ICalProp** to **InnoDB** (by dropping it and allowing it to repopulate). | No comments. |

## 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, " Database Schema Version Changes for Calendar Server Beginning with Release 7 Update 2" 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.<br><br>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.<br><br>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.<br><br>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.)<br><br>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.<br><br>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.<br><br>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 dependences on Java, MySQL Server, Oracle GlassFish Server, and other items for Calendar Server.

*Table 9–4 Calendar Server Upgrade Dependencies*

| Upgrade Dependency | Introduced in Which Calendar Server Version? |
|---|---|
| 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 Communications Suite **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 7.0.5

This section describes how to upgrade to Calendar Server 7.0.5. Depending on your current Calendar Server version, some steps are not required.

## Prerequisites for Upgrading Calendar Server

1. Download the necessary software.
   a. Download the Calendar Server software and Directory Server setup script, **comm_dssetup**, either as part of the media pack for Oracle Communications Unified Communications Suite, or as a patch.
      – Download the Oracle Communications Unified Communications Suite distribution from the Oracle software delivery website.
      – Download a Calendar Server 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`
2. 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/CS7U5**), and extract the Communications Suite 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 GlassFish Server, on the Calendar Server front-end hosts, create a temporary directory (for example, **/temp/GF3**), and extract the GlassFish 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, " Database Schema Version Changes for Calendar Server Beginning with Release 7 Update 2" 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

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.

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

For information about installing Java, see the *Installing Java for Communications Suite* documentation at:

`https://wikis.oracle.com/display/CommSuite/`

Notes:

- 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 7.0.5, 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 the topic on Communications Suite Directory Server setup in *Communications Suite Installation Guide* at:

   https://wikis.oracle.com/display/CommSuite

## Running commpkg upgrade

This section assumes that you have previously put the Communications Suite distribution 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 about running the installer, see the **commpkg upgrade** usage documentation at:

   https://wikis.oracle.com/display/CommSuite

   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 Communications Suite software distribution 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 Communications Suite 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.

      ```
      cd CommunicationsSuite_home
      ./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 GlassFish 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 Communications Suite Installer and choosing. If you are applying a patch, see the instructions in "Installing Patches" then return to this chapter to resume the upgrade process.

```
cd CommunicationsSuite_home
./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 GlassFish 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 GlassFish 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=mypassword --all-databases > myDataDump.sql
```

Also, you must stop the Calendar Server front-end services by shutting down the GlassFish 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 GlassFish 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.

## 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))
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:

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 6 and 7 Coexistent Deployment

This chapter describes how to set up Oracle Communications Calendar Server 7 in an existing Sun Java Calendar Server 6 deployment. In this environment, you have some users who have migrated to Calendar Server 7 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 Calendar Server 7 or Calendar Server 6. They do not have calendars on both Calendar Server 7 and Calendar Server 6.

For more information on performing a Calendar Server 7 migration, see "Migrating to Calendar Server."

Topics:



## Calendar Coexistence Overview

For coexistence to work, each calendar server needs to be able to determine if users are Calendar Server 6 or Calendar Server 7 users.

On Calendar Server 7 servers:

- Users are identified as Calendar Server 7 users by the presence in their Directory Server LDAP entry of a particular attribute which is configured as described in "Configuring the Calendar Server 7 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 Calendar Server 7 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 Calendar Server 7 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 Calendar Server 7 to use **davuniqueid** as the unique identifier, you must add the new unique identifier not only to the users migrated to Calendar Server 7, 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 Calendar Server 7, see "Migrating to Calendar Server."

## Minimum Software Requirements

- Calendar Server 6.3: At least patch level 44.
- Calendar Server 7: At least Calendar Server 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 Calendar Server 7 server's free/busy URL. If the Calendar Server 7 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 Calendar Server 7 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 a Calendar Server 7 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 6.3 users no longer log in by mistake and cause calendars to be autocreated.

## Configuring the Calendar Server 7 Environment

Perform the following steps on the Calendar Server 7 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 a Calendar Server 7 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 Calendar Server 7 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

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 Calendar Server 7 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 Calendar Server 7 Configuration

When a scheduling invitation or free/busy request comes to a Calendar Server 7 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 Calendar Server 7 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 Calendar Server 7 host. If a user is on a Calendar Server 6 host, for scheduling an invitation, an iMIP invitation is sent by the Calendar Server 7 host.

# 11

# Migrating to Calendar Server

This chapter describes how to migrate from Sun Java Calendar Server 6 to Oracle Communications Calendar Server 7.



## Introduction to Migrating

Because the database formats differ between Calendar Server 6 and Oracle Communications Calendar Server (formerly known as Sun Java System Calendar Server 7), 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 Calendar Server 7 hosts. For more information, see "Creating a Calendar Server 6 and 7 Coexistent Deployment."

---

## How the Migration Works

The migration utility performs the following sequence of events:

1. Communicates with the Calendar Server 7 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 Calendar Server 7 back-end database, as specified by the LDAP **davStore** attribute. The fetched data is massaged to fit the format for the Calendar Server 7 host before being stored. Different mapping helper functions perform the data massaging.

6. The user's calendar is autocreated 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 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 Calendar Server 7 because it was previously migrated or was explicitly created in Calendar Server 7, 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.

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 LDAP. Resource calendars are not migrated by just migrating the resource owner, but by explicitly migrating the resource account itself.
- Resource Owner: In Calendar Server 7 Update 3, the migration sets the owner of a resource 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 Calendar Server 7, 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 Calendar Server configuration parameters in *Calendar Server System Administrator's Guide*.

## High-Level Migration Steps

Migrating from Calendar Server 6 to Calendar Server 7 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***^a^r^g** 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 Calendar Server 7 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 Calendar Server 7 host to use.

   **Note:** Not setting this attribute means the back-end host is local to the Calendar Server 7 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 Calendar Server 7 deployment.

## Migrating From an SSL-Enabled Calendar Server 6 Deployment

For a Calendar Server 7 host to communicate with a Calendar Server 6 host over SSL, the Calendar Server 7 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 Calendar Server 7 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 Calendar Server 7 host and import it into the Java runtime environment that is used by Calendar Server 7.

   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 Calendar Server 7 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

The count reported by the Calendar Server 7 **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 6.3 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 Calendar Server 7 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 6.3 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 6.3 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 6.3 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 6.3 configuration parameters as previously described.
2. Stop and restart calendar services on the Calendar Server 6.3 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/00000000000000000000000000000040
5ce44e000063da0000003100004aa0/ while storing:
00000000000000000000000000000405ce44e000063da0000003100004aa0
```

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 :
0000000000000000000000000000000486c05b900001c6b000001c400000532: 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@iplan
et.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 Calendar Server 7, delete the account that you migrated and run the migration again.

## 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

```
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/OLEXS1CSATTACHIAA
A007110202XS1CSCLIENTIAAAec798b438d5469da5af862534819cf2XS1CSFILENAME/ENCODEIN//BA
SE64VALU//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

**Important:** 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 might want to disable Oracle GlassFish Server from servicing incoming requests.

   See "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

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 might want to disable GlassFish Server from servicing incoming requests.

   See "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

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 might want to disable GlassFish Server from servicing incoming requests.

   See "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.
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.en
abled=true
server-config.network-config.network-listeners.network-listener.http-listener-1.en
abled=true
Command set executed successfully.
```

# A

# 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 A–1 describes the **init-config** options.

***Table A–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*. |

## 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 A–2 describes the **config-mysql** *options*:

*Table A–2 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 A–3 describes the **config-oracle** *options*:

***Table A–3 config-oracle Options***

| Option | Description |
|---|---|
| **--help** \| **-?** | Prints help. |
| **--calendar** \| **-c** | Configures the calendar database. |

*Table A–3 (Cont.) config-oracle Options*

| Option | Description |
|---|---|
| **--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 A–4 describes the **populate-davuniqueid** options:

***Table A–4 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* ]<br>(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=icscalendaruser)(objectclass=icscalendarresource)(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 Calendar Server 7 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:
```