Oracle® Communications Calendar Server

Concepts

Release 7.0.5

E54933-01

February 2015

ORACLE®

Oracle Communications Calendar Server Concepts, Release 7.0.5

E54933-01

Contents

Preface v

Audience v
Related Documents v
Documentation Accessibility v

1 Overview of Calendar Server

About Calendar Server 1-1
Calendar Server Software Architecture 1-1
Calendar Server Servlets and Core 1-2
Calendar Server Back-End Database 1-2
Calendar Server Quality of Service 1-3
Security 1-3
Horizontal and Vertical Scalability 1-3
High Availability 1-3
Directory Server Integration 1-3

2 Calendar Server 6 and 7 Comparison and Coexistence

Overview of Calendar Server 7 and Calendar Server 6 2-1
Calendar Server Architecture and Services 2-1
Coexistence Between Calendar Server 6 and Calendar Server 7 2-2
Installation and Configuration 2-2
Data Migration 2-3
Special User Accounts 2-3
Proxy Administrative Accounts 2-3
End-User Administration 2-3
Data Formats, Protocols, and Standards 2-3
Access Control 2-4
Public APIs 2-4
Backup and Restore 2-4
Log Files 2-5
Maintaining Configuration Files 2-5
Notifications and Reminders 2-5
How Free/Busy is Calculated for All Day Events 2-6

3 Calendar Server and Directory Server Integration

How Does Calendar Server Use Directory Server? 3-1
LDAP Schema Used by Calendar Server 3-1
Calendar Server 6 Versus Calendar Server 7 LDAP Schema Usage 3-2
Similarities and Differences Between Calendar Server 6 and Calendar Server 7 Schema 3-2
Deprecated Attributes and Object Classes in Calendar Server 7 3-2
Special LDAP Object Classes for Calendar Server 3-3
icsCalendarUser 3-4
icsCalendarResource 3-4
icsCalendarDomain 3-4
davEntity 3-4
groupofUniqueNames 3-5
Important Attributes for Calendar Server 3-5
Unique ID Attribute 3-5
Mail Attribute 3-6
Status Attribute 3-6
Store ID Attribute 3-6
External Authentication Attributes 3-7
Indexing the Directory Server 3-7
Special Calendar Server Users and Groups 3-8
About the Calendar Server Proxy User 3-8
Calendar Server Administrator ID 3-9
LDAP Updates During Configuration 3-9
How Calendar Server Authenticates with Directory Server 3-9
Background Information 3-9
First Step: Domain Lookup on UG Server 3-10
Second Step: User Authentication 3-10
User Authentication Default Behavior 3-10
Credentials Verification: Authentication Server 3-10
Alternative Behavior Using External Directory Authentication 3-11
Third Step: User Info Lookup on UG Server 3-11
Alternative Behavior Using External Directory Authentication 3-11
Sample Calendar Server User LDIF 3-12

4 Calendar Server Supported Standards

Preface

This guide provides information on the concepts associated with Oracle Communications Calendar Server.

Audience

This document is intended for system administrators whose responsibility includes Calendar Server. This guide assumes you are familiar with the following topics:

- Basic administrative procedures for your platform operating system
- Oracle GlassFish Server
- Lightweight Directory Access Protocol (LDAP), if you plan to use an LDAP directory server to store user information
- Oracle Directory Server Enterprise Edition
- System administration and networking
- General deployment architectures

Related Documents

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

- *Calendar Server System Administrator's Guide*: Provides instructions for administering Calendar Server.
- *Calendar Server Installation and Configuration Guide*: Provides instructions for installing and configuring Calendar Server.
- *Calendar Server Release Notes*: Describes the new features, 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

Overview of Calendar Server

This chapter provides an introduction to Oracle Communications Calendar Server.

About Calendar Server

Calendar Server (also referred to as Calendar Server 7 and formerly known as Calendar Server for CALDAV Clients and Sun Java System Calendar Server) is a carrier-grade, highly scalable, secure, and reliable calendaring and scheduling platform. Calendar Server is compliant with the latest calendaring and scheduling standards, including the CalDAV access protocol, which makes it usable with Apple iCal, iPhone, Thunderbird Lightning, and any other CalDAV client.

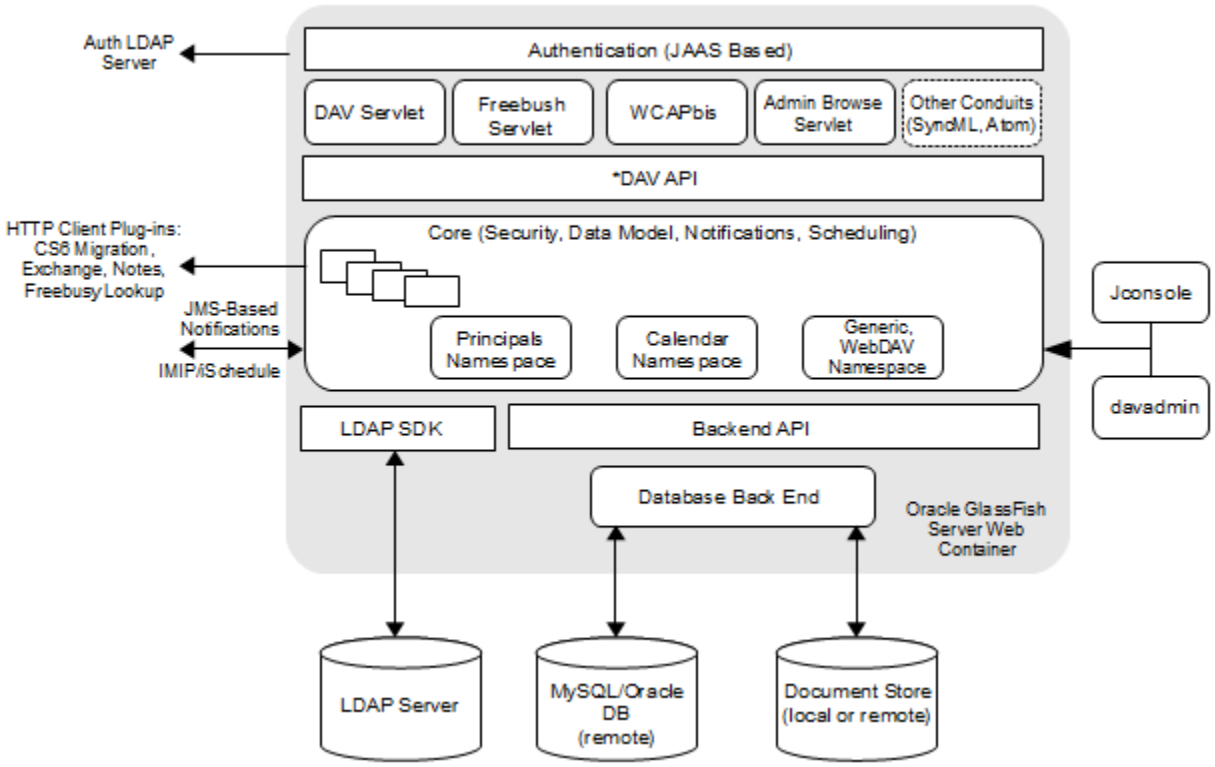
Calendar Server provides many calendaring and scheduling capabilities, including:

- Personal appointments (one-time/recurring) and reminders
- Multiple calendars per user (Work calendar, Home calendar)
- Document store for storing event/task attachments
- Multiple access points, including desktop clients (Apple iCal, Outlook, Lightning), Convergence web client, and mobile clients (iPhone, Android)
- Availability checks
- Invitation notifications
- Special handling of resource scheduling
- Comprehensive access control settings including settings for groups
- Sharing and subscription of calendars

Calendar Server Software Architecture

Figure 1–1 represents the Calendar Server software architecture.

Figure 1–1 Calendar Server Software Architecture



This figure shows the following Calendar Server components:

- Servlets
- Core
- Back ends

The following sections explain these components in more detail.

Calendar Server Servlets and Core

Calendar Server consists of a collection of servlets and core components, both deployed in a GlassFish Server web container. The servlets provide HTTP services, including a simple freebusy service and Java Management Extensions (JMX) interface for the **davadmin** command. The core components handle security, data management, notifications, and scheduling tasks.

Calendar Server Back-End Database

Calendar Server stores user data in a back-end database, which can be either MySQL Server or Oracle Database. In addition to this user database, Calendar Server uses the document store to store large data in the Calendar Server, such as calendar events and todos with large attachments. CalDAV and WCAP clients are able to retrieve these attachments over HTTP. Calendar Server requires one document store per configured Calendar Server back-end database host, unless the database itself is Oracle Database, which you can configure to contain large data.

Calendar Server Quality of Service

End users demand secure, always-on, always-available services. From an IT perspective, downtime increases costs. Calendar Server is a robust calendar solution designed to deliver the quality of service necessary to meet the demands of end users.

Security

Calendar Server comes with built-in security features. Certificate-based authentication is available for use in secure environments. Domain-level access controls provide more flexibility for communications service providers (CSPs) hosting multiple enterprises. Virus scanning of data and DoS prevention help keep the environment secure.

Horizontal and Vertical Scalability

The lower the number of servers required to deliver services to larger communities, the easier it is to manage those servers, maintain quality of service, and keep costs down. Calendar Server is based on a scalable architecture that helps maximize hardware investment by enabling it to handle a high volume of simultaneous real-time calendaring events.

High Availability

Calendar Server, built on Oracle GlassFish Server, leverages the application server clustering capabilities. The architecture uses a standard MySQL back end, which means that tools do not need to be reinvented for administering the database. Because the database is based on MySQL, it leverages existing MySQL clustering solutions to support continuous availability and centralized cluster management.

Directory Server Integration

Calendar Server is tightly integrated with Oracle Directory Server Enterprise Edition. Integration with the Lightweight Directory Access Protocol (LDAP)–based directory server enables the central management and storage of user and account information, simplifying management and administration.

2

Calendar Server 6 and 7 Comparison and Coexistence

This chapter discusses the differences between Oracle Communications Sun Calendar Server (formerly Sun Java System Calendar Server 6) and Oracle Communications Calendar Server (also referred to as Calendar Server 7), as well as the issues for a coexistent deployment.

Overview of Calendar Server 7 and Calendar Server 6

Calendar Server 7 is Oracle's next-generation Calendar Server, which supports the new standard for calendar access called Calendaring Extensions to WebDAV (CalDAV).

Calendar Server 7 is a newly designed server and replaces the existing Calendar Server 6. Calendar Server 7 operates in a very different manner from Calendar Server 6. Though both servers use the standard iCal format for data, Calendar Server 6 supports only Web Calendar Access Protocol (WCAP) for data access, making it usable only with Oracle-supplied clients and connectors. Calendar Server 7 supports the standard CalDAV access protocol, which makes it usable with Apple iCal, iPhone, Thunderbird Lightning, and any other CalDAV client.

Calendar Server Architecture and Services

Calendar Server 6 was designed as a stand-alone application that consisted of multiple processes, including:

- **csadmin**: Manages alarm notifications and group scheduling requests
- **cshttpd**: Listens for HTTP commands from Calendar Server end users, receives the user commands, and returns calendar data
- **csstored**: Creates automatic backups of the calendar database
- **csnotifyd**: Sends notifications of events and to-do's (tasks)
- **csdwpd**: Manages calendar databases spread across multiple back-end servers within the same Calendar Server configuration

Calendar Server 7 is a servlet that you deploy into a web container (GlassFish Server). Thus, Calendar Server 7 does not contain its own start and stop programs, unlike Calendar Server 6. (You start and stop Calendar Server 7 through the GlassFish Server **asadmin** command.) The administrative commands have been completely rewritten as well for Calendar Server 7. The Calendar Server 7 commands do not have one-to-one mapping to the Calendar Server 6 commands.

For more information about administering Calendar Server 7 by using the **davadmin** command, see the command-line utilities topic in *Calendar Server System Administrator's Guide*.

Calendar Server 6 used an embedded Berkeley DB database while Calendar Server 7 uses a separate database, which can be either MySQL Server or Oracle Database. Though you must install and maintain this database outside of Calendar Server, maintenance is eased by using the database-specific tools.

Coexistence Between Calendar Server 6 and Calendar Server 7

You can install Calendar Server 6 and Calendar Server 7 on the same host. Each server does not use the other's data. You can ensure communication of free-busy information and iCalendar Message-Based Interoperability Protocol (iMIP) scheduling between the two servers by using the setup described in *Calendar Server Installation and Configuration Guide*. Also, this setup requires that you install the latest Calendar Server 6 patch (patch-ID 121659).

Installation and Configuration

Table 2–1 shows the installation and configuration information relating to the different Calendar Server versions.

Table 2–1 Calendar Server Installation and Configuration Differences by Version

Version	Description
Calendar Server 6.3	Installing and configuring Calendar Server 6.3 consists of the following high-level steps: 1. Installing the software by using the Communication Suite 6 Installer. 2. Running the Communications Suite 6 Directory Server Setup script, **comm_dssetup.pl**, to configure Directory Server. 3. Running the Calendar Server configuration program (**csconfigurator.sh**) to initially configure your site's specific requirements and to create a new **ics.conf** configuration file. 4. Customizing your system by editing parameters in the **ics.conf** file.
Calendar Server 7	Installing and configuring Calendar Server 7 consists of the following high-level steps: 1. Installing and configuring the GlassFish Server as the web container. 2. Installing either MySQL Server or Oracle Database software and setting up the database. 3. Running the Communications Suite Directory Server setup script, **comm_dssetup.pl**, to configure Directory Server. 4. Installing the Calendar Server 7 software by using the Communications Suite installer. 5. Running the Calendar Server 7 configuration script (**init-config**) to initially configure your site's specific requirements.

For more information, see *Calendar Server Installation and Configuration Guide*.

Data Migration

See the topic on migrating from Sun Java System Calendar Server 6 to Calendar Server 7 in *Calendar Server Installation and Configuration Guide*.

Special User Accounts

Calendar Server 6.3 special accounts include the following:

- Calendar Server Administrator (**calmaster**) Account
- Superuser (**root**)
- Non-root User (**icsuser**, **icsgroup**)

Calendar Server 7 special accounts include the following:

- Calendar Server Administrator (**calmaster**) Account
- Calendar Server User and Group Accounts
- Superuser (**root**)
- **mysql** user and group
- GlassFish Server Administrator Account

Proxy Administrative Accounts

In Calendar Server 6.3, to enable administrators to administer user calendars, the default value for the following parameter in the **ics.conf** file is set as shown:

```
service.http.allowadminproxy="yes"
```

There is no equivalent in Calendar Server 7. That is, you cannot disable administrative access to calendar users.

End-User Administration

Administrators with the proper permissions can add, delete, or modify user LDAP entries or resource LDAP entries by using the Delegated Administrator Utility (command line) or Console (GUI). Calendar Server 7 requires Delegated Administrator 7. Calendar Server 6.3 uses Delegated Administrator 6.4.

Additionally, when necessary, you can use **ldapmodify** to modify LDAP entries directly.

Both servers support Schema 1 and Schema 2. For more information, see *Communications Suite Schema Reference*.

Calendar Server 7 requires the **davEntity** object class to set up multiple back-end hosts for horizontal scalability. Calendar Server 7 does not support the Database Wire Protocol (DWP), which was used for distributing calendar data across multiple back-end hosts in Calendar Server 6.3.

Data Formats, Protocols, and Standards

The Calendar Server 6.3 data format is modeled after RFC 5545, Internet Calendaring and Scheduling Core Object Specification (iCalendar).

Calendar Server 6.3 supports Web Calendar Access Protocol (WCAP) 3.0.

The main interface used by Calendar Server 7 to interact with the data repository is the CalDAV protocol, which is based on HTTP. Thus, any client that supports CalDAV, such as Apple iCal and Thunderbird/Lightning, can access Calendar Server 7. Calendar Server 7 supports a newer version of the WCAP protocol, with no guarantee of backward compatibility.

Additionally, Calendar Server 7 supports the server-to-server iSchedule protocol, which means that it can also act as a client. The server also exposes the following HTTP-based services:

- Simple free-busy service
- Administrative browse UI

Finally, Calendar Server 7 offers a JMX-based interface to query and alter the repository. This interface is primarily used by the **davadmin** command-line utility but can also be used by standard clients, such as Jconsole, and by custom JMX clients, either locally or remotely.

Access Control

Calendar Server 6.3 uses Access Control Lists (ACLs) to determine the access control for calendars, calendar properties, and calendar components such as events and to-do's (tasks).

Calendar Server 7 supports a similar but more user friendly, rich access control mechanism. See the topic on administering Calendar Server access in *Calendar Server System Administrator's Guide* for more information.

Calendar Server 6.3 also supports access control settings at domain levels control inter-domain access in a multi-domain setup.

Calendar Server 7 supports similar functionality but the ACL format is slightly different and the LDAP attribute used is different from that used by Calendar Server 6.3. See the topic on managing domain access controls in *Calendar Server Security Guide* for more information.

Public APIs

For Calendar Server 6.3, see the following:

- WCAP 3.0: See *Sun Java System Calendar Server 6.3 WCAP Developer's Guide.*
- ENS: See *Communications Suite Event Notification Service Guide*.

For Calendar Server 7, see *Calendar Server WCAP Developer's Guide*.

Backup and Restore

When properly configured, the Calendar Server 6.3 **csstored** service creates automatic backups of the calendar database. You can configure Calendar Server 6.3 for automatic backups either during initial configuration or at a later time.

In Calendar Server 7, you use the **davadmin db** command to back up and restore the back-end database. There is no automatic database function as there is in Calendar Server 6.3. You can write **cron** jobs for Calendar Server 7 deployments to perform incremental backups by using the **davadmin db backup** command.

For more information, see the topic on best practices for backing up and restoring databases in *Calendar Server System Administrator's Guide.*

Log Files

Calendar Server 6.3 maintains the following log files:

- **admin**
- **dwp**
- **http**
- **httpd access**
- **notify**
- **store**

The default log location is **/var/opt/SUNWics5/logs**, which you can modify.

Calendar Server 7 maintains the following log files:

- **commands**
- **errors**
- **scheduling**
- **telemetry**
- **scan**

Each log file has its own configuration attribute that controls the log file location, maximum size, and log level. The default log file location is the logs directory, under the **data** directory that you enter during initial configuration. You can set the logging level for Calendar Server logs either by using the **davadmin** command or in the GlassFish Server logs by using the GlassFish Server Administration Console.

For more information, see the topic on administering logging in *Calendar Server System Administrator's Guide*.

Maintaining Configuration Files

You change the Calendar Server 6.3 configuration by adding parameters to or modifying existing parameters in the **ics.conf** file. The **ics.conf** file is located in the following directory:

- Oracle Solaris: **/etc/opt/SUNWics5/cal/config**
- Red Hat Linux: **/etc/opt/sun/calendar/config**

You change the Calendar Server 7 configuration by using the **davadmin config modify** command.

Notifications and Reminders

Calendar Server 6 can be configured to use an external generic service called the Event Notification Service (ENS), which accepts reports of server-level events that can be categorized into specific areas of interest. This service also notifies other servers that have registered interest in certain categories of events. Calendar Server 6 uses ENS to send and receive alarm notifications that include the creation, deletion, or modification of calendar events and tasks, as well as general operational warning and error messages.

Note: The Calendar Server 6.3 software also contains support for Java Message Queue for notification, but **csnotifyd** does not subscribe to it. Thus, it is not part of the default alarms and notification system. For more information, refer to the Sun Java System Java Message Queue documentation.

Calendar Server 7 provides Java Message Service (JMS) notifications and email notifications for database changes and event or task email alarms. You can configure the server to produce JMS notifications for every database change and every alarm. If you choose, you can write your own subscribers to these notifications. In addition, Calendar Server 7 provides a subscriber program, which you can configure, that consumes the JMS notifications and sends email for database changes and email alarms.

How Free/Busy is Calculated for All Day Events

All events are considered to calculate busy time unless explicitly marked as transparent. The time blocked corresponds to the event timings in the event's time zone. For floating events (events with no associated time zone) and all day events, the busy time is calculated in the calendar's time zone. For example if there is a floating event from 12 pm to 1 pm, for lunch, in a calendar whose time zone is America/Los_Angeles, for free/busy calculation purposes, the event would be considered as a 12 pm to 1 pm event in the America/Los_Angeles time zone.

3

Calendar Server and Directory Server Integration

This chapter describes how Oracle Communications Calendar Server uses Oracle Directory Server Enterprise Edition, what LDAP schema is necessary, and what specific object classes and attributes are required. Where appropriate, differences between Calendar Server 7 and Calendar Server 6 are described.

How Does Calendar Server Use Directory Server?

Calendar Server uses Directory Server to store and access LDAP data for individual users, groups, and domains. The LDAP data is used in a variety of ways, including:

- Authenticating users
- Determining a user's, group's, or domain's status
- Administering domain level access control
- Retrieving important information, such as the user's unique ID, email address, calendar store back-end ID, and so on
- Retrieving default values for various user properties

You must install and provision Directory Server prior to installing and configuring Calendar Server. You also need to prepare the Directory Server LDAP schema by running the **comm_dssetup.pl** script, which is provided as part of the Unified Communications Suite installer. This script prepares the LDAP directory by adding the necessary Communications Suite schema. For more information, see the topic on **comm_dssetup.pl** in *Communications Suite Installation Guide* at:

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

For information on installing and configuring Calendar Server, see *Calendar Server Installation and Configuration Guide*. In addition, perform post-configuration steps to enable certain levels of LDAP searches required for calendar searches and subscription, and to enable a secure connection between Calendar Server and Directory Server. See the topic on post-configuration tasks in *Calendar Server Installation and Configuration Guide* for more information.

LDAP Schema Used by Calendar Server

To understand the schema that is used by Calendar Server, refer to *Communications Suite Schema Reference*. Specific LDAP object classes support Calendar Server 7.

The **davcore.ldapattr.*** configuration parameters govern the default attributes and object classes used by the Calendar Server 7. Default values are set based on the

Communications Suite schema. See the topic on configuration parameters in *Calendar Server System Administrator's Guide* for more information.

Calendar Server 6 Versus Calendar Server 7 LDAP Schema Usage

This section contains the following topics:

- Similarities and Differences Between Calendar Server 6 and Calendar Server 7 Schema
- Deprecated Attributes and Object Classes in Calendar Server 7

Similarities and Differences Between Calendar Server 6 and Calendar Server 7 Schema

In Calendar Server 6, a user's LDAP entry requires the **icsCalendarUser** object class for calendaring to work for that user. Unless disallowed by the server configuration, Calendar Server 6 automatically provisions users by adding the **icsCalendarUser** object class to LDAP, creating the database entries for the user, and thus enables calendaring for the user upon first login or invite. Basic provisioning of the user entry in the LDAP directory is required for this to work.

Unlike Calendar Server 6, Calendar Server 7 does not add or modify LDAP data nor does it require the presence of a particular LDAP object class like **icsCalendarUser**. Unless disallowed by the server configuration, Calendar Server 7 automatically creates the database entries for a user upon initial login or invite, thus enabling the user. Basic provisioning of the user entry in the LDAP directory is required for this to work.

In Calendar Server 6, you can use both the **icsStatus** or **icsAllowedService** attributes to enable or disable calendaring service. In Calendar Server 7, you use only the **icsStatus** attribute to enable or disable the service. For more information, see the information on **icsStatus** in *Communications Suite Schema Reference*.

In Calendar Server 6, the **icsDWPHost** attribute specifies the user's back-end data store. In Calendar Server 7, it is the **davstore** attribute.

Deprecated Attributes and Object Classes in Calendar Server 7

The following Communications Suite LDAP attributes used by Calendar Server 6 are no longer used by Calendar Server 7. These attributes are from the **icsCalendarUser**, **icsCalendarResource**, and **icsCalendarDomain** object classes:

- **icsAllowedServiceAccess**
- **icsCalendar**
- **icsCalendarOwned**
- **icsDefaultSet**
- **icsDWPHost**
- **icsExtended**
- **icsExtendedUserPrefs** (but still used by Convergence)
- **icsFirstDay** (but still used by Convergence)
- **icsFreeBusy**
- **icsGeo**
- **icsPartition**

- **icsPreferredHost**
- **icsQuota**
- **icsSet**
- **icsSubscribed**
- **icsTimezone**
- **nswcalDisallowAccess**
- **aclGroupAddr**
- **icsAlias**
- **icsCapacity**
- **icsContact**
- **icsExtended**
- **icsExtendedResourcePrefs**
- **icsSecondaryowners**
- **icsDefaultacl**
- **icsAllowRights**
- **icsAnonymousAllowWrite**
- **icsAnonymousCalendar**
- **icsAnonymousDefaultSet**
- **icsAnonymousLogin**
- **icsAnonymousSet**
- **icsDWPBackEndHosts**
- **icsExtendedDomainPrefs**
- **icsDefaultAccess**
- **icsDomainAllowed**
- **icsDomainNotAllowed**
- **icsMandatorySubscribed**
- **icsMandatoryView**
- **icsRecurrenceBound**
- **icsRecurrenceDate**
- **icsSessionTimeout**
- **icsSourceHtml**

The **icsCalendarGroup**, **icsAdministrator**, and **icsCalendarDWPHost** object classes are also no longer used by Calendar Server 7.

This information can also be found in *Communications Suite Schema Reference*.

Special LDAP Object Classes for Calendar Server

This section describes the following object classes:

- icsCalendarUser
- icsCalendarResource
- icsCalendarDomain
- davEntity
- groupofUniqueNames

icsCalendarUser

This object class defines a user entry with calendar service. While addition of this object class for calendar users is recommended, it is not mandatory. By default, a user entry that contains the deployment's "Unique ID Attribute" (a unique identifier in the form of an LDAP attribute whose value is used to map each calendar account to a unique account in the Calendar Server database), and **uid**, **password**, and **mail** attributes, works as a valid calendar user entry. The **icsCalendarUser** object class gives you more flexibility in enabling and disabling calendaring for provisioned users by using the **icsStatus** attribute. To require addition of this object class for a user entry to be considered as a valid calendar user entry, set the value of the Calendar Server configuration parameter **davcore.ldapattr.userobject** to **icsCalendarUser**.

See *Calendar Server System Administrator's Guide* for more information on using the **davcore.ldapattr.userobject** configuration parameter.

icsCalendarResource

This object class is required to define a resource entry with calendar service. A resource in the scheduling context is any shared entity that can be scheduled by a calendar user, but does not control its own attendance status.

To use a different object class instead, set the custom value for the **davcore.ldapattr.resourceobject** configuration parameter. The **icsCalendarResource** object class provides the owner attribute that by default specifies the owner of the resource.

For more on resource accounts, see the topic on administering resource calendars in *Calendar Server System Administrator's Guide*.

icsCalendarDomain

This object class defines a domain entry with calendar-enabled users. It enables setting domain-level access control for calendar users' cross-domain access by using its **icsDomainNames** and **icsDomainAcl** attributes. See the topic on managing domain access controls in *Calendar Server Security Guide* for details. You can use the **icsStatus** attribute in this object class to enable or disable calendaring service for an entire domain.

davEntity

This is a common object class that you can add to any of the others such as **icsCalendarUser**, **icsCalendarResource**, or **icsCalendarDomain**, to include some commonly needed attributes. This object class includes the **davStore** attribute that defines the back-end calendar store host for a user, or an entire domain in a multiple back-end setup. It also includes the **davUniqueID** attribute to specify a globally unique ID for any LDAP entry.

groupofUniqueNames

The Communications Suite schema does not define any specific group object classes. By default, Calendar Server considers entries with **groupofuniquenames**, **groupofurls**, or **inetmailgroup** object classes as groups. This is defined by the Calendar Server **davcore.ldapattr.groupobject** configuration parameter.

Group entries make it easy to invite a whole set of users, or to set access control for a whole set of users. Groups are identified by using their **mail** attribute value. To determine members of a group, Calendar Server uses LDAP attributes defined by the following configuration parameters:

- **davcore.ldapattr.dngroupmember**: Defines members by using their distinguished name (**dn**); default value is **uniquemember**
- **davcore.ldapattr.mailgroupmember**: Defines members by using their email address; default value is **mgrprfc822mailmember**
- **davcore.ldapattr.urlgroupmember**: Defines members by using a URL; default value is **memberurl**

> **Note:** Both **uniquemember** and **memberurl** are considered for checking ACLs while **mgrprfc822mailmember** is not. Thus, members of a **uniquemember** or **memberurl** group can read, write, subscribe, and so forth, to a calendar depending on the ACL. In the majority of cases, use **uniqueMember** when configuring a calendar group. **mgrprfc822mailmember** is application specific and useful to invite users that are not part of your deployment.

For more information, see the topics on inviting LDAP groups and managing dynamic group ACLs in *Calendar Server System Administrator's Guide*.

Important Attributes for Calendar Server

This section contains the following topics:

- Unique ID Attribute
- Mail Attribute
- Status Attribute
- Store ID Attribute
- External Authentication Attributes
- Indexing the Directory Server

Unique ID Attribute

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.

The **nsUniqueId** attribute was initially chosen to be used by Calendar Server for a unique ID. However, the **nsUniqueId** attribute is no longer recommended because the value cannot be preserved if the LDAP entry must be moved or recreated. The **nsUniqueId** attribute does guarantee that the value is unique throughout the Directory Server instance but unfortunately that is not enough. The recommendation

is now to choose an attribute whose value can be guaranteed to be unique. In addition, you should add checking to your provisioning process and tools to ensure that the chosen unique ID attribute is defined for every entry and is indeed unique.

You can use the Calendar Server supplied **davUniqueId** attribute to define a unique ID for any LDAP entry. It is recommended that it be used as the value of the **davcore.uriinfo.permanentuniqueid** parameter. This parameter defines which attribute Calendar Server uses as the unique identifier for users, groups, and resources. Calendar Server also provides the **populate-davuniqueid** script to set this attribute's value for all existing LDAP entries to a unique value.

In the Calendar Server database, 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.

Mail Attribute

By default, Calendar Server uses the **mail** attribute. This attribute is used for many important functions, such as the default account identifier for the **davadmin** command-line utility, the address to be used for scheduling, the default address for email notifications, and so on. The Calendar Server database also stores the value of this attribute. If scheduling is performed by using other mail attributes, such as **mailAlternateAddresses**, the values are canonicalized to the mail value before storing in the database. If you change the value of this attribute, you must perform additional steps to changing it in the LDAP directory. See the topic on changing a user's email address in the Calendar Server database in *Calendar Server System Administrator's Guide* for more information.

This attribute is also required for resource entries. Though resources do not receive email, Calendar Server uses this address value to identify and schedule the resource. Hence this value should be unique to the resource, and other values such as the owner's email address should not be used. For more information, see the topic on managing a resource calendar's mailbox in *Calendar Server System Administrator's Guide*.

Status Attribute

By default, Calendar Server uses the **icsStatus** attribute to enable or disable a user for calendaring services. Absence of this attribute or a value of **active** indicates active status. Values of **removed**, **deleted**, or **inactive** disable the service. Any other value may also enable the service but is not recommended.

If a calendar account's LDAP **icsStatus** attribute is populated and is not set to **active**, the account is not searched nor are any results fetched for that account when running Calendar Server **davadmin** or WCAP commands. That is, Calendar Server returns search results only for active accounts and does not return unusable data such as inactive calendars.

Store ID Attribute

By default, Calendar Server uses the **davStore** attribute, which indicates the back-end host that stores a user's data if the deployment is configured for multiple back ends. Also, if the deployment uses the Convergence client or has a Calendar Server 6 and Calendar Server 7 co-deployment setup, this attribute is required. In the latter two

cases, if there are no multiple Calendar Server 7 back-end hosts, then the value used by default must be **defaultbackend**.

If your deployment consists of multiple back ends, you must use one of the **store.dav.***xx***.backendid** values previously configured to set as the value for the **davStore** attribute. You must explicitly provision this attribute on your back-end configuration. Neither Calendar Server nor Delegated Administrator automatically provision this attribute.

Once a user is active, do not change the **davStore** attribute value. To move users to another back-end store, see the topic on moving calendar users in *Calendar Server System Administrator's Guide*.

External Authentication Attributes

The attributes described in this section support authentication against an external Directory Server. For more information, see the topic on configuring external authentication in *Calendar Server System Administrator's Guide*.

- **externalAuthPreUrlTemplate**: This attribute is used for authentication when using external Directory Servers. It is used to set the LDAP URL that defines how users must be searched for in the external Directory Server against which authentication is performed. This attribute belongs to the **inetDomainAuthInfo** object class.
- **externalAuthPostUrlTemplate**: This attribute is used for finding the internal Directory Server entry for a user authenticated by using external Directory Servers. It is used to set the LDAP URL that must be used to map the external Directory Server authenticated user to a user in the internal Directory. This attribute belongs to the **inetDomainAuthInfo** object class.

Indexing the Directory Server

Certain attributes need to be indexed for presence, equality, or sub-string search for better performance. Some of this indexing is done by Directory Server itself or by the **comm_dssetup** script. Use Table 3–1 to decide if you have the correct indexes. Use the **dsconf list-indexes** command to list the current indexes. (See the topic on listing indexes in *Sun Directory Server Enterprise Edition 7.0 Administration Guide*). Use the **create-index** command for creating new indexes. (See the topic on creating indexes in *Sun Directory Server Enterprise Edition 7.0 Administration Guide*).

Table 3–1 Attributes for Indexing

Attribute	Index
associatedDomain	eq, pres
cn	eq, pres, sub
inetDomainBaseDN	eq, pres
mail	eq, pres, sub
mailAlternateAddress	eq, pres
member	eq
memberOf	eq, pres, sub
objectClass	eq
owner	eq
sunPreferredDomain	eq, pres

Table 3–1 (Cont.) Attributes for Indexing

Attribute	Index
uid	eq
uniqueMember	eq

If you are concerned about LDAP performance, check the LDAP access logs for the entry **notes=U** to find unindexed searches. For example:

```
[15/Feb/2012:06:45:11 -0800] conn=110405 op=13 msgId=21 - SRCH
base="o=example.com,o=dav" scope=2 filter="(|(uid=cal*)(cn=*cal*)(mail=*cal*))"
attrs="cn davStore icsStatus mail mailAlternateAddress nsUniqueId owner
preferredLanguage uid objectClass isMemberOf uniqueMember memberURL
mgrpRFC822MailMember kind"

[15/Feb/2012:06:45:17 -0800] conn=110405 op=13 msgId=21 - RESULT err=4 tag=101
nentries=1000 etime=6 notes=U
```

Special Calendar Server Users and Groups

Calendar Server creates two special users during the initial configuration, one for serving as a proxy to the Directory Server, and the other for acting as the administrator ID for Calendar Server itself.

Topics in this section:

- About the Calendar Server Proxy User
- Calendar Server Administrator ID
- LDAP Updates During Configuration

About the Calendar Server Proxy User

Calendar Server uses a proxy user to bind to the Directory Server when making requests. This user belongs to the Calendar End User Administrators Group, which has proxy rights. This special Calendar Server user makes Directory Server requests on behalf of the end user for whom the request is being carried out. The proxy process takes into account the Directory ACIs for that particular end user. The DN (Distinguished Name) of this newly created user is added to the server configuration as the **base.ldapinfo.ugldap.binddn**, for example:

- Server configuration option:

  ```
  base.ldapinfo.ugldap.binddn=uid=cal-admin-sca-peanut.example.com-20111102184916
  Z,ou=People,o=example.com,o=isp
  ```

- Sample LDIF file used to create that user and group in the Directory Server:

  ```
  dn: cn=Calendar End User Administrators Group,
   ou=Groups, o=isp
  changetype: modify
  add: uniqueMember
  uniqueMember: uid=cal-admin-sca-peanut.example.com-20111102184916Z,
   ou=People, o=example.com,o=isp

  dn: o=isp
  changetype: modify
  add: aci
  aci: (target="ldap:///o=isp")
  ```

```
 (targetattr="*")
 (version 3.0; acl "Calendar Server End User Administrator Proxy Rights -
product=davserver,schema 2 support,class=admin,num=1,version=1"; allow (proxy)
groupdn="ldap:///cn=Calendar End User Administrators Group, ou=Groups, o=isp";)
```

Calendar Server Administrator ID

The other special user that Calendar Server creates is the Service Administrator user, by default, the **calmaster** user. The Calendar Server **base.ldapinfo.serviceadminsgroupdn** configuration parameter specifies the name of an LDAP group for which membership in the group means super user privileges as far Calendar Server is concerned. The **calmaster** user belongs to this Service Administrators group and thus has full access to all users' calendaring information. Convergence always authenticates and proxies on behalf of the end user to Calendar Server as **calmaster**. You can add additional users to the Service Administrators group if you require that more users have these privileges.

LDAP Updates During Configuration

The Calendar Server initial configuration script, **init-config**, creates the LDAP tree's base containers and the default domain if it does not already exist. It also sets up the special administrative groups and users previously mentioned. ACIs are also added to give read, search, and proxy rights to the administrative users. An ACI is also added to give read and search ACIs to all authenticated users.

You may also want to enable users to search for other users whose calendars they can subscribe to. This requires permission to search for other users in LDAP. If such a permission is not already granted, you must add an ACI to the user/group suffix in Directory Server. For more information, see the topic on adding LDAP access control in *Calendar Server Installation and Configuration Guide*.

How Calendar Server Authenticates with Directory Server

The following information describes how Directory Server authenticates a Calendar Server user by using basic user ID and password authentication.

Topics in this section:

- Background Information
- First Step: Domain Lookup on UG Server
- Second Step: User Authentication
- Third Step: User Info Lookup on UG Server

Background Information

Most requests to Directory Server are made against the user/group LDAP server (as defined in the **base.ldapinfo.ugldap.*** Calendar Server configuration parameters). Only the bind request to check the user's credentials is issued against the authentication LDAP server (as defined in the **base.ldapinfo.authldap.*** configuration parameters).

When an external directory is configured for the user's domain, a search is first issued against that external directory. The bind request to check the user's credentials is also issued against that external directory. Most of the information discovered through this process (including a hash of the logged-in user passwords) is cached in memory (with a configurable time-to-live value).

First Step: Domain Lookup on UG Server

Consider the following search request:

```
Search Request: base DN = o=dav, scope = 2, filter =
&(objectClass=sunManagedOrganization)(|(sunPreferredDomain=example.com)(associated
Domain=example.com))
```

The domain lookup is performed as follows:

1. The base DN (the top-level of the Directory Server tree) for this search is determined by the **base.ldapinfo.dcroot** Calendar Server configuration parameter.
2. The domain value is extracted from the user-provided user ID by using the **base.ldapinfo.loginseparator** Calendar Server configuration parameter. For example, if **arnaudq@example.com** is the user ID and the login separator character is "@," the domain is determined to be **example.com**).
3. Alternatively, if no login separator is found in the user ID, the domain is assumed from the **base.ldapinfo.defaultdomain** Calendar Server configuration parameter. An LDAP lookup is still issued in that case.

Second Step: User Authentication

The default behavior is to use the user/group server and the authentication server for authentication.

If the domain entry retrieved in the previous step has an **externalAuthPreUrlTemplate** LDAP attribute, the user authentication is performed against an external directory.

User Authentication Default Behavior

To illustrate how the user (user/group server) authentication is performed, consider the following search request:

```
Search Request: base DN = o=example.com,o=dav, scope = 2, filter = uid=arnaudq
```

The base DN for this search is extracted from the domain entry retrieved during the initial domain lookup (with some differences between Schema 1 and Schema 2). Next, the search filter is constructed either from the **inetDomainSearchFilter** LDAP attribute of the domain entry or from the **base.ldapinfo.searchfilter** configuration parameter. In both cases, the configuration value is based on the following template:

- **%U**: Name part of the login name (that is, everything before the login separator stored in the servers configuration)
- **%V**: Domain part of the login string
- **%o**: Original login ID entered by the user (for example, **uid=%U**)

This lookup should return an LDAP entry containing, in addition to a DN, the list of attributes defined by the **base.ldapinfo.userattrs** configuration parameters (the default is **mail** and **ismemberof**).

Credentials Verification: Authentication Server

Consider the following bind request:

```
Bind Request: version = 3, DN = uid=arnaudq,ou=people,o=example.com,o=dav
```

Unlike the other requests, this one is made by using the authentication LDAP Server. The DN used is the one retrieved during the authentication user lookup step and the password is the one provided by the end user. Assuming that the bind request is

successful, the LDAP entry returned during the authentication user lookup step is used to:

1. Check whether the corresponding user belongs to the administrative group.
2. Extract the mail attribute of the entry (the attribute name itself is configurable through the **davcore.ldapattr.mail** Calendar Server configuration parameter). Pay attention to this configuration parameter, as it is also used in different other places.

The authentication ends at this point. A set of principal objects containing the mail value (and optionally administrator privileges) are constructed as a result.

Alternative Behavior Using External Directory Authentication

The **externalAuthPreUrlTemplate** LDAP attribute contains an LDAP URL defining what external directory server to use, and the type of search to issue to find the user within that directory. The host name part of the URL contains some identifier pointing to an actual LDAP Pool defined in the server configuration. The search filter in that LDAP URL is also a template, containing the same keywords as previously described (**%o**, **%U**, and **%V**). Once the entry is found, the credential verification (as previously described) is performed by using that entry's DN. The process of configuring external authentication for a particular domain is described by the topic on configuring external authentication in *Calendar Server System Administrator's Guide*.

Third Step: User Info Lookup on UG Server

Consider the following search request:

```
Search Request: (base DN = o=example.com,o=dav, scope = 2, filter =
|(mail=arnaud.quillaud@example.com)(mailalternateaddress=arnaud.quillaud@example.c
om)
```

While not exposed, externally, the set of (JAAS) authentication modules is configurable. The domain lookup, authentication user lookup, and credentials verification steps correspond to the basic LDAP Auth module but other modules can be plugged in. The contract between an Auth Module and the server is that after successful authentication, the mail attribute of the principal must be made available to the server. Once the mail attribute value is extracted, a new lookup based on that email address is issued. This lookup is part of the core server processing.

The base DN for this search is extracted from the domain part of the email address (as described in "First Step: Domain Lookup on UG Server" although this information is already most likely cached at that point). The search filter is constructed from the **davcore.uriinfo.emailsearchfiltertemplate** configuration parameter (the default value is **|(mail=%s)(mailalternateaddress=%s)**). The returned entry is used to retrieve user information. The list of requested attributes is controlled by the **davcore.uriinfo.subjectattributes** configuration parameter.

Alternative Behavior Using External Directory Authentication

Once the external directory has authenticated the user, a mapping needs to happen between that entry and the corresponding entry in the Communications Suite directory.

If all external directory user entries have a **mail** attribute value corresponding to their Communications Suite equivalent entry, no further configuration is required that is different from the internal authentication setup described previously. At the "Second Step: User Authentication", the list of attributes to be retrieved must simply include the **mail** attribute.

If no such direct mapping exists, an LDAP search is issued against the internal Communications Suite directory. This second search is defined by the **externalAuthPostUrlTemplate** domain entry attribute. Like **externalAuthPreUrlTemplate**, the **externalAuthPostUrlTemplate** attribute contains an LDAP URL defining what type of search to issue to find the user within that directory. In this case, a server name is not required and should not be defined, as the lookup is done against the Communications Suite directory. The search filter can be a template or fixed filter. They can contain the patterns previously mentioned, as well as the **%A** *attributename_* pattern, which is substituted with the first LDAP attribute value retrieved from the external authentication directory in the "Second Step: User Authentication". This pattern may appear multiple times with different attribute names. Of course, those attributes must be part of the list of attributes to retrieve in the LDAP URL.

For a given login ID and domain, those patterns are substituted for their actual values before the search is issued. The search is then conducted, the correct entry is found, and the authentication is considered successful.

See *Communications Suite Schema Reference* for more information on the **externalAuthPostUrlTemplate** and **externalAuthPreUrlTemplate** attributes.

Sample Calendar Server User LDIF

The following sample LDIF file is for a user that has access to Calendar Server. The user was created in Delegated Administrator. Delegated Administrator does not distinguish between Calendar Server 6 and Calendar Server 7 users. It includes object classes from both Calendar Server 6 (**icsCalendarUser**) and Calendar Server 7 (**davEntity**).

```
dn: uid=caluser1, ou=People, o=example.com, o=dav
sn: NS
givenName: CalUser
inetUserStatus: active
icsStatus: active
userPassword: {SSHA}GFxOCTbnS90zfWmqmvTgc51gOK36AOUCqXzZRA==
cn: caluser1
uid: caluser1
objectClass: sunUCPreferences
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: inetUser
objectClass: ipUser
objectClass: inetmailuser
objectClass: icscalendaruser
objectClass: inetlocalmailrecipient
objectClass: userpresenceprofile
objectClass: daventity
davUniqueId: c4df2181-422a11dd-8002d2ed-c263972e
mail: caluser1@example.com
mailAlternateAddress: caluser1@mail.domain1.example.com
mailHost: mail.example.com
mailUserStatus: active
mailDeliveryOption: mailbox
```

4

Calendar Server Supported Standards

This chapter lists national, international, industry and de-facto standards related to electronic calendaring and for which support is claimed by Oracle Communications Calendar Server. Most of these are Internet standards, published by the RFC Editor and approved by the Internet Engineering Task Force (IETF). Standards for documents from other sources are noted.

Table 4–1 shows the RFC documents that are relevant to national and international standards for calendaring.

Table 4–1 RFC Calendaring Documents

Document	Description
RFC2616	Hypertext Transfer Protocol HTTP/1.1
RFC4791	Calendaring Extensions to WebDAV (CalDAV)
RFC5545	Internet Calendaring and Scheduling Core Object Specification
RFC5546	iCalendar Transport-Independent Interoperability Protocol (iTIP)
RFC6047	iCalendar Message-Based Interoperability Protocol
RFC6578	Collection Synchronization for WebDAV
RFC6638	Scheduling Extensions to CalDAV

Table 4–2 shows the documents that are in draft state.

Table 4–2 Draft Documents

Document	Description
caldav-ctag-02	Calendar Collection Entity Tag (CTag) in CalDAV
draft-daboo-srv-caldav-10	Locating CalDAV and CardDAV services