Oracle® Communications Contacts Server

Installation and Configuration Guide

Release 8.0

E50286-05

May 2021

ORACLE

Oracle Communications Contacts Server Installation and Configuration Guide, Release 8.0

E50286-05

Contents

4 Contacts Server Pre-Installation Tasks

5 Installing Contacts Server

10 Upgrading Contacts Server

11 Uninstalling Contacts Server

12 Installing Patches

A commpkg Reference

B comm_dssetup.pl Reference

C Contacts Server Configuration Scripts

Preface

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

Audience

This document is intended for system administrators or software technicians who install and configure Contacts 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 or Oracle WebLogic Server
- Oracle Directory Server Enterprise Edition and LDAP
- System administration and networking

Related Documents

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

- *Contacts Server System Administrator's Guide:* Provides instructions for administering Contacts Server.
- *Contacts Server Release Notes*: Describes the known issues and required third-party products and licensing.
- *Contacts Server Security Guide*: Provides guidelines and recommendations for setting up Contacts Server in a secure configuration.
- *Contacts Server RESTful Protocol Guide*: Describes the RESTful protocol that enables HTTP clients to fetch, add, and edit address book related data that is stored by Contacts Server.

Nomenclature

The following nomenclature is used throughout the document.

Convention	Meaning
Application Server	The term **Application Server** or **application server** is used in this document to refer to either GlassFish Server or WebLogic Server. **Supported Application Server**: Oracle Communications Contacts Server 8.0.0.4.0 and previous releases were deployed on GlassFish Server, which is no longer supported by Oracle. For that reason, Contacts Server 8.0.0.5.0 and beyond are only supported on Oracle WebLogic Server. Oracle strongly recommends that you upgrade your Contacts Server environments to release 8.0.0.5.0 or higher and migrate to WebLogic Server to receive full Oracle support.

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

Contacts Server Installation Overview

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

Overview of Contacts Server Installed Components

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

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

Contacts 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 Contacts Server to use notifications, you must also have an email server installed, such as Oracle Communications Messaging Server.

Overview of the Contacts 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 Contacts Server is dependent, including:
   - Java
   - Application 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 Calendar Server media pack.
5. Install and configure Contacts Server.
6. (Optional) Configure additional Contacts 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 Contacts Server is installed, you might perform additional security-related tasks. For more information, see *Contacts Server Security Guide*.

Contacts Server Installation Options

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

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

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

Ensuring a Successful Contacts Server Installation

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

Follow these guidelines:

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

Directory Placeholders Used in This Guide

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

Table 1–1 Contacts Server Directory Placeholders

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

2

Planning Your Contacts Server Installation

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

About Contacts Server

Contacts Server enables end users to store and retrieve contact information such as name, email address, photo, birthdays, and any other information that relates to the contact. Contacts Server supports all properties defined in the vCard specification, RFC 6350, available at:

http://tools.ietf.org/html/rfc6350

Contacts Server provides a Network Address Book that facilitates centralized storage and access of contacts for a large number of users. Being full-featured, it not only provides contact creation, management and searching capabilities along with multiple group and multiple address book support, but includes features that enterprises demand, such as Global Address List integration and address book sharing. Contacts Server provides the capability to back up, synchronize, and merge address books in a secure, accessible, reliable, and device independent way. Contacts Server also enables end users to create multiple address books for organizing and sharing contacts.

Contacts Server Front-End and Back-End Components

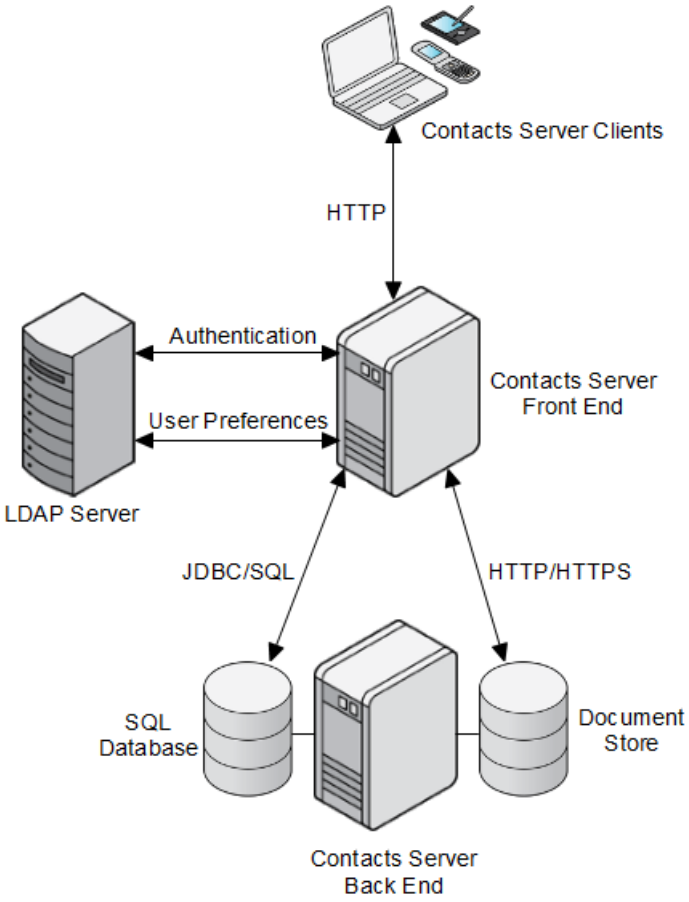
Contacts Server consists of the following front-end and back-end components:

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

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

Figure 2–1 shows a Contacts Server configuration that uses two hosts.

Figure 2–1 Contacts Server Front-End and Back-End Configuration



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

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

Planning Your Contacts Server Installation

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

- Using a Single Web Container for Multiple Applications
- Deciding on the Database
- Planning for Multiple Contacts Server Hosts
- Planning the Document Store
- Planning for Virus Scanning
- Planning for the Unique Identifier

Using a Single Web Container for Multiple Applications

Contacts Server requires that you use the application server as its web container. For production deployments, deploy Contacts Server to the root context (/) to simplify configuring mobile clients. The URI syntax is:

`http://`*`contacts-server-host-name`*`:`*`port`*`/`*`contacts-server-webapp`*`/dav/principals/`*`email-address`*`/`

For example:

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

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

You cannot deploy Contacts Server and Calendar Server in the same domain, even if you use different contexts, for example, **/nabserver** and **/davserver**. You must deploy Contacts Server and Calendar Server in different domains.

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

Deciding on the Database

You can use either MySQL Server or Oracle Database as the Contacts 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 Contacts Server Hosts

Using multiple Contacts Server hosts can help you:

- Avoid network latency and unnecessary bandwidth consumption by positioning the server closer to the client (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 Contacts 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 Contacts Server With Multiple Hosts" for more information.

Planning the Document Store

The Contacts Server document store is used to store and retrieve *large data*, such as photos and logos. You must set up one document store per configured Contacts 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 Contacts Server instantiation. For example, the local document store could be part of a Contacts 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, Contacts 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, Contacts Server accesses the documents by using either HTTP or HTTPS. If you use Oracle Database, you can configure it to contain both the contact 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 contacts data and the document store.

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

Planning for Virus Scanning

Contacts Server supports virus scanning of attachments, such as photos. If you choose to configure virus scanning, decide whether to use an existing Oracle Communications Messaging Server MTA, or to deploy a dedicated MTA-only Messaging Server installation to scan for viruses. To use virus scanning for Contacts 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*.

Though this information is written for Calendar Server, it also applies to Contacts Server.

Planning for the Unique Identifier

Contacts 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 contacts database (the MySQL Server or Oracle Database that stores the Contacts 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 Contacts 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 Contacts Server and start using it. Contacts Server provides the **davuniqueid** attribute, which is the recommended attribute to use. For more information, see the unique identifier documentation in *Calendar Server Concepts*. Though this information is written for Calendar Server, it also applies to Contacts Server.

System Deployment Planning

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

- Planning for High Availability
- Planning Backup Strategies

Planning for High Availability

You can configure Contacts Server front-end and back-end hosts to be highly available. For high availability of front-end hosts, deploy the hosts behind a load balancer. See "Using Load Balancing" for more information. Choices for making the Contacts Server back-end database highly available include MySQL asynchronous replication and Oracle Data Guard. Refer to the documentation for those products for installation and configuration instructions. You can also configure the document store for high availability.

Using Load Balancing

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

Planning Backup Strategies

Backing up and restoring data is one of the most important administrative tasks for your Contacts Server deployment. You must implement a backup and restore policy for your Contacts 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 Contacts Server data are:

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

For more information, see the backup and restore best practices topic in *Contacts Server System Administrator's Guide*.

Contacts Server Logical Architecture

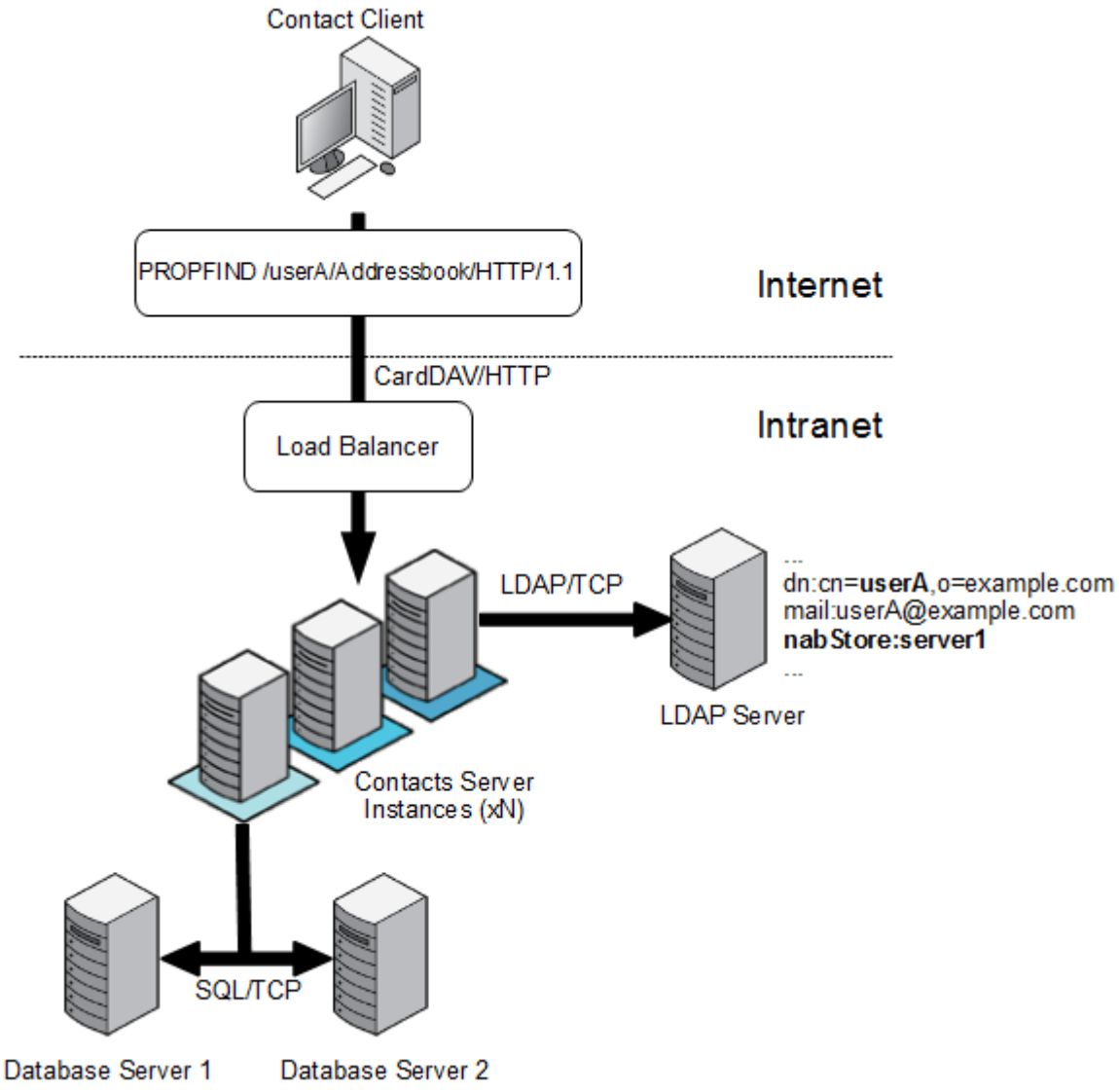
When planning your Contacts Server logical architecture, you can use the following options:

- **Single-tiered Contacts Server architecture:** You can deploy all components on a single host.
- **Two-tiered Contacts Server architecture:** You can deploy Contacts Server with the front-end component installed on a separate host and the database back end installed on another host.
- **Two-tiered, multiple server Contacts 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 Contacts Server Physical Architecture

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

Figure 2–2 Contacts Server Physical Architecture



About Installing a Secure System

You can configure Secure Sockets Layer (SSL) between the Contacts Server application server front-end hosts and database back-end hosts. You can also configure SSL between the Contacts Server application 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 servers for SSL by configuring either the required **trustStore** files or wallets. Then you configure the Contacts Server front-end hosts to connect over SSL by making use of the stored certificates.

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

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

3

Contacts Server System Requirements

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

Software Requirements

Contacts Server is installed on the application server domain. It uses either an Oracle database or MySQL Server database for data storage.

Supported Operating Systems

Table 3–1 lists operating systems that support Contacts Server.

Table 3–1 Contacts Server Operating System Requirements

Operating System	CPU	Required Patches
Solaris OS 11	SPARC, x64	See the Solaris 11 Release and Installation documentation for more information.
Oracle Linux 6 and Red Hat Enterprise Linux 6 64-bit	x64	See the Oracle Linux documentation and Red Hat Enterprise Linux documentation for more information.

Required Software

Table 3–2 lists the database software choices for Contacts Server.

Table 3–2 Contacts Server Database Requirements

Product	Version
Oracle Database	11g Release 2, 12c To upgrade to Oracle Database 12c from Oracle Database 11g, see ***Oracle Database Upgrade Guide*** at: `https://docs.oracle.com/database/121/UPGRD/toc.htm` **Note**: Contacts Server Patch 8.0.0.5.0 is certified with Oracle Database 19c.

Table 3–2 (Cont.) Contacts Server Database Requirements

Product	Version
MySQL Server	5.5.40, 5.6.22, 5.7.26 To upgrade to MySQL Server 5.6 from MySQL Server 5.5, see "Upgrading from MySQL 5.5 to 5.6" at: http://dev.mysql.com/doc/refman/5.6/en/upgrading-from-previous-series.html To know the changes in MySQL 5.7, see “Configuration Changes” at: https://dev.mysql.com/doc/refman/5.7/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 Contacts Server.

Table 3–3 Contacts Server Software Requirements

Product	Version	Notes
Oracle Directory Server Enterprise Edition (formerly Sun Java System Directory Server)	Oracle Directory Server Enterprise Edition 6.x, 7, 11.x	For a fresh installation, use the latest version of Directory Server 11gR1.
Application Server	GlassFish Server 3.1.2.x WebLogic Server 12.2.1.3.0 and 12.2.1.4.0.	Required as a web container: For GlassFish Server, download the patch from My Oracle Support at: https://support.oracle.com For WebLogic Server, download the generic installer from http://edelivery.oracle.com For WebLogic Server 12.2.1.4.0, the installer is available as **Oracle Fusion Middleware 12c (12.2.1.4.0) WebLogic Server and Coherence** and contains **fmw_12.2.1.4.0_wls.jar** and **fmw_12214_readme.html**. For example, for Linux Platform, you should download the **V983364-01.zip** package. **Supported Application Server**: Oracle Communications Contacts Server 8.0.0.4.0 and previous releases were deployed on GlassFish Server, which is no longer supported by Oracle. For that reason, Contacts Server 8.0.0.5.0 and beyond are only supported on Oracle WebLogic Server. Oracle strongly recommends that you upgrade your Contacts Server environments to release 8.0.0.5.0 or higher and migrate to WebLogic Server to receive full Oracle support.

Table 3–3 (Cont.) Contacts Server Software Requirements

Product	Version	Notes
Java	For GlassFish Server, use Java 7 For WebLogic Server, use Java 8	The Java version is according to the application server that you have selected to deploy Contacts Server. **Note**: You must download the latest security patch updates available for the respective Java version.

Client Requirements

Contacts Server supports any standard CardDAV client. Table 3–4 shows which clients were tested with Contacts Server.

Table 3–4 Contacts Server Clients

Client	Tested Version
CardDAV-Sync	0.4.5
Apple Mac OS client	8.0 (1365)
Apple iOS client	7.0.2 (11A501)

Hardware Requirements

The number and configuration of the systems that you employ for your Contacts 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 Contacts Server deployed on a single managed server in a WebLogic or GlassFish domain.

Table 3–5 Contacts Server Minimum Hardware Requirements

Component	Requirement
Disk Space	Approximately 20 MB required for Contacts Server software.
RAM	8 GB

Information Requirements

During Contacts 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.

Contacts Server Information

Table 3–6 lists the Contacts Server information that you provide during initial configuration.

Table 3–6 Contacts Server Information

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

Table 3–6 (Cont.) Contacts Server Information

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

Database Information

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

Table 3–7 Back-End Database Information

Information Type	Default Value
The type of Contacts 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**
Contacts Server database user	**nab**
Contacts Server database user password	No default value.
Contacts Server database name	**nab**

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*
Contacts Server database user	**nab**
Contacts 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 Contacts Server back end with which the document store is associated.	**defaultbackend**
The path to the Contacts Server document store.	**/var/opt/sun/comms/nabserver/db**
(Remote document store only) The name of the host where the remote Contacts Server document store is located.	No default value.
(Remote document store only) The port number for the remote Contacts Server document store.	**8008**

GlassFish Server Information

Table 3–11 lists the GlassFish Server information that you provide during initial configuration.

Table 3–11 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.
URI path of the deployed server	**https://***FQDN of host***:443/** (root directory) For information about using a **.well-known** URI, such that access to **/** (root) or **/.well-known/carddav/** is redirected to the **/dav/principals/** URI. For more information, refer to the Enabling CalDAV and CardDAV Autodiscovery section in *Calendar Server System Administrator's Guide*.

WebLogic Server Information

Table 3–12 lists the Oracle WebLogic Server information that you provide during initial configuration.

Table 3–12 WebLogic Server Information

Information Type	Description
WebLogic Server installation directory	Directory in which WebLogic Server is installed. For example, **WLS_HOME/Oracle_Home**
WebLogic Server domain directory	The directory in which domain directories are created. For example: **WLS_HOME/Oracle_Home/user_projects/domains/**domain1.
WebLogic Server document root directory	The WebLogic Server document root directory. For example: **WLS_HOME/Oracle_Home/user_projects/domains**/domain1.
WebLogic Server target instance name	The name of the WebLogic Server's Managed Server target name. For example: server1
WebLogic Server virtual server	The name of the WebLogic Server's Managed Server target name. For example: server1

Table 3–12 (Cont.) WebLogic Server Information

Information Type	Description
WebLogic Server administration server host	FQDN of host
Is administration server port secure	Whether Oracle WebLogic Server administration server port is running over SSL. Default: Enabled
WebLogic Server administration server port	**7001** This is the port with which you log in to WebLogic Administration Server. **Note**: If you set **Is administration server port secure** to **true**, this must be the SSL port of WebLogic Administration Server.
WebLogic Server administrator user	The admin user name to log in to WebLogic Server Administration Server.
WebLogic Server administrator user password	The password to log in to WebLogic Server Administration Server.
URI path of the deployed server	https://FQDN of host:7003/davserver Pattern should be https://*FQDN of host*:*ManagedServer_Port context-root* ManagedServer_Port: WebLogic Server domain must be set up with at least one ManagedServer which hosts Contacts Server. You must provide the port of that Managed Server here. **Note**: If you select Secure mode, ensure to enter **https** and the SSL port of that Managed Server instance. **context-root**: The default **/davserver** is an example of the context to deploy Contacts Server. For more information, refer to the Enabling CalDAV and CardDAV Autodiscovery section 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 depends on the previous value and the user/group suffix of the Directory Server.
Contacts Server administrator user	**nabmaster**
Contacts 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

Contacts Server Pre-Installation Tasks

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

Pre-installation and configuration tasks include:

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

Installing Java

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

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

Download Java from the Oracle website:

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

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

Installing GlassFish Server

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

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

Installing and Configuring WebLogic Server

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

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

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

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

To install and configure Oracle WebLogic Server for Contacts Server:

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

   See the following Oracle WebLogic Server resources for more information:

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

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

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

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

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

     ```
     DERBY_FLAG="false"
     ```

     export DERBY_FLAG

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

   cd *WebLogic_Domain*/**bin**

   source ./**setDomainEnv.sh** or . ./**setDomainEnv.sh**

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

   To enable SSL and configure keystores in Oracle WebLogic Server:

   - See the Oracle WebLogic Server documentation at:

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

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

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

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

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

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

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

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

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

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

```
Weblogic_Domain/config
```

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

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

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

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

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

Installing Directory Server

Contacts 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 Contacts 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 Calendar Server media pack, adds the necessary Communications Suite schema to the LDAP. See "Preparing Directory Server" for more information.

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

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

Installing the Database

You can use either Oracle Database or MySQL Server as the Contacts Server database (the contacts 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 Contacts Server. When you configure Contacts 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 Contacts 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 Contacts Server

Installing and Configuring MySQL Server

To install and configure MySQL Server:

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

   http://support.oracle.com

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

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

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

   - On Solaris:

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

   - On 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 need to 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 ***Contacts 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 the Contacts Server Software" section.

Installing and Creating an Oracle Database Instance for Contacts Server

You can install an Oracle Database for Contacts 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 Contacts Server Schema

> **Note:** The Oracle Database instance must be a Unicode database, that is, the database character set must be AL32UTF8. Contacts 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.
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 the "Installing the Contacts Server Software" section.

Creating a New Database From an Existing Oracle Installation

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

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

- Oracle Database 12c:

  https://docs.oracle.com/database/121/ADMIN/create.htm#ADMIN002

- Oracle Database 11g, Release 2:

  http://docs.oracle.com/cd/E11882_01/server.112/e10897/install.htm#ADMQS0232

2. When creating the database instance, follow these guidelines:
   - When choosing the template, select the default **General Purpose**.
   - When choosing the database management method in the Select Database Management Options window, select **Use Database Control for Database Management**.
   - When choosing the storage options, select **File System**.
   - When choosing database content, deselect the sample schema.
   - When selecting the database character set in the Specify Database Configuration Options window, select **UTF-8 AL32UTF8**.

Continue with the "Installing the Contacts Server Software" section.

Using an Existing Database for the Contacts Server Schema

You can use an existing Oracle database for the Contacts 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 Contacts Server after installing Contacts 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 Contacts Server schema. Continue to "Installing Contacts 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 "Installing a New Oracle Database".

5

Installing Contacts Server

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

Before installing Contacts Server, read these chapters:

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

Installation Assumptions

The instructions in this chapter assume:

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

Installing Contacts Server

The tasks to install Contacts Server are as follows:

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

Downloading the Contacts Server Software

To download Contacts Server software:

1. Download the Contacts Server software, and the Oracle Communications Directory Server Setup **comm_dssetup.pl** script, for your operating system from the Oracle software delivery website, located at:

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

   The Contacts Server server software is part of the Calendar Server media pack.

2. Copy the Contacts Server ZIP file to a temporary directory on your Contacts Server hosts and extract the files.
3. Copy the Directory Server Setup ZIP file to a temporary directory on your Directory Server hosts and extract the files, to be able to install and run the **comm_dssetup.pl** script.

Preparing Directory Server

You prepare your Directory Server by running the **comm_dssetup.pl** script against it. You can run the **comm_dssetup.pl** script in either interactive or silent mode. For silent mode instructions, see "Running the comm_dssetup.pl Script in Silent Mode".

Running the comm_dssetup.pl Script in Interactive Mode

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

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

   ```
   commpkg install
   ```

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

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

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

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

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

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

Installing the Contacts Server Software

To install the Contacts Server software:

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

   ```
   commpkg install
   ```

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

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

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

Preparing MySQL Server for Use with Contacts 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 Contacts Server. See "Running the config-mysql Script" for more information about running the **config-mysql** script.

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 Contacts Server. Do not use a different version.

To prepare MySQL Server for use with Contacts Server:

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

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

2. Create a **mysql** user by running 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 database.

The following example uses **nab** as the MySQL user name, and **nab** as the database name. You can use different names.

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

For more information, see the MySQL CREATE USER syntax at:

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

Replace *localhost* with the Contacts 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 **nab** 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 you need to 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 information about how to configure MySQL to run with SMF at:

      https://dev.mysql.com/doc/refman/5.7/en/solaris-installation.html

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

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

Preparing Oracle Database for Use with Contacts 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 Contacts Server software, copy the **config-oracle** script and the **Util.pm** module, located in the *ContactsServer_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 ContactsServer_home/tools/unsupported/bin
   config-oracle -c
   ```

4. Respond to the following prompts.

   ```
   Enter the Oracle SYS user password:
   Enter the name of the contacts database user (nab):
   Enter the password for contacts database user <user>:
   Please enter password again:
   ```

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

After preparing Oracle Database for use with Contacts Server, continue with "Configuring Contacts Server."

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 Contacts 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 Contacts Server:

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

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

Installing Contacts Server in Silent Mode

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

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

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

This section includes:

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

Performing a Contacts Server Silent Installation

To perform a Contacts Server silent installation:

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

     ```
     commpkg install --dry-run
     ```

2. Copy the state file to each host machine and edit the file as needed. See "Silent Mode File Format".

3. Run the silent installation on each host. For example:

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

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

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

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

About Upgrading Shared Components

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

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

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

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

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

Silent Mode File Format

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

Table 5–1 Silent Mode File Options

Option	Description	Example
VERB	Indicates which function to perform. For a silent install, this is set to **install**.	VERB=install
ALTDISTROPATH	Indicates an alternate distro path.	ALTDISTROPATH=SunOS5.10_i86pc_DBG.OBJ/release

Table 5–1 (Cont.) Silent Mode File Options

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

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

Installing Contacts Server on Solaris Zones

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

The topics in this section include:

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

Installing on Solaris OS Zones: Best Practices

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

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

  Install shared components and OS patches in the global zone that are to be shared among all zones. However, do not install and run products from the global zone.

- Use whole root non-global zones to run Contacts Server.

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

Be aware of the following zone aspects:

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

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

Installing into a Non-Global Whole Root Zone

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

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

Installing into a Non-Global Sparse Root Zone

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

1. Follow the pre-installation requirements as described in "Contacts Server Pre-Installation Tasks".

2. Verify that you are about to install the shared components and OS patches in the global zone and not the sparse root zone. To verify you are in the global zone, run **zonename**. The output should be global.
3. Run the installer in the global zone and only install or upgrade the OS patches and the Shared Components. Do not install Contacts Server in the global zone. To do this, add a ~ (tilde) to the component number you want to install in the sparse zone.

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

Configuring Contacts Server

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

Contacts Server Initial Configuration Prerequisites

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

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

Validating and Storing Oracle WebLogic Server SSL Details

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

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

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

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

   ```
   cd Weblogic_Domain/bin
   source ./setDomainEnv.sh   OR  . ./setDomainEnv.sh
   ```

2. Set the **WLST_PROPERTIES** environment variable depending on the selected Oracle WebLogic Server keystore configuration.

- If the **CustomIdentityandCustomTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

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

  where *WebLogic_home/user_projects/domains/base_domain/mytrust.jks* is the location of truststore file.

- If the **CustomIdentityandJavaStandardTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_ PROPERTIES** variable to:

  ```
  export WLST_
  PROPERTIES="-Dweblogic.security.TrustKeyStore=JavaStandardTrust"
  ```

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

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

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

Configuring Java for Contacts Server

Do the following on the Contacts Server front-end hosts so that Contacts Server locates the proper JDK:

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

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

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

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

   ```
   export JAVA_HOME=JDK_location
   ```

If you use WebLogic Server:

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

Running the Contacts Server Initial Configuration Script

To run the Contacts Server initial configuration:

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

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

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

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

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

   See "Contacts Server Configuration Scripts" for more information.

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

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

   Note: If you do not use the **-W** option, by default, GlassFish Server is used to configure the **init-config** script.

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

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

   Choose MySQL database or Oracle database (the default is **mysql**), then enter the following information, depending on whether you chose MySQL database or Oracle database:

   - MySQL Database Details
     - Server Host
     - Server Port
     - Contacts Database User
     - Contacts Database User Password (The password you provided while creating the **nab** user in "Preparing MySQL Server for Use with Contacts Server".)
     - Contacts Server DB Name

- Oracle Database Server Details
  - Server Host
  - Server Port
  - Service Name
  - Contacts DB User
  - Password (The password you provided while creating the **nab** user in "Preparing Oracle Database for Use with Contacts Server".)

  If you receive a database connection error, you are prompted to re-enter all the database information in this step. When the validation passes, continue with the next step.

7. Enter the Contacts Server Back-End Document Store Settings.
   - Back-end document store type, either **local**, **dbdocstore**, or **remote** (See "Configuring the Contacts Server Document Store" for more information.)
   - Back-end name to with which the document store is associated
   - Back-end store path
8. Enter the application server configuration details.
   - Install Directory
   - Domain Directory
   - Document Root Directory
   - Server Target Name
   - Virtual Server Identifier
   - Application server administration server host
   - Secure Administration Server Instance
   - Application server administration server port
   - Administrator User ID
   - Administrator Password
   - Full URL 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. The following example URIs describe the behavior of this entry:
     - **https:/host.example.com:8080/** - Deploy in root.
     - **https:/host.example.com:8080/nabserver** - Deploy in **nabserver**.
     - **https:/host.example.com:8080** - Not allowed, deployment context cannot be blank.
     - **https:/host.example.com:443/nab/dav** - Not allowed, deployment context cannot be more than one level.
   - If you use GlassFish Server:

     Using context root as the URI path replaces the GlassFish Server's main index.html file. Therefore, the GlassFish Server welcome page does not display at http://host:port.

- If you use WebLogic Server:

  The URI pattern for deploying the server must be:

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

  where:

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

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

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

  Note: You cannot deploy Contacts Server and Calendar Server in the same domain, even if you use different contexts, for example, **/nabserver** and **/davserver**. You must deploy Contacts Server and Calendar Server in different domains.

9. Enter the User/Group Directory Server Details.

   - LDAP URL

     The default uses LDAP over SSL (`ldaps://`). If you want Contacts Server to communicate by using SSL with Directory Server in the runtime configuration, you need to use LDAPS in the URL specification. You must also put the certificate database containing the Directory Server certificate in the *ContactsServer_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 can enable LDAP SSL communication in the runtime configuration as described in *Contacts 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 contacts 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 ran 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 ran 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.

- Contacts Server administrator user
- Contacts 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 Contacts Server, answer **Y**.

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

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

You can run the **init-config** script in a silent mode for GlassFish Server and WebLogic Server.

If you use GlassFish Server, see "Contacts Server Configuration Scripts" for more information.

If you use WebLogic Server, obtain the correct statefile to run using init-config in a silent mode. You should not use the statefile from the past release versions.

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

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

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

  ContactsServer_home/**install**/*nabserver-config-timestamp*/**saveState**

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

About the new parameter: From the Contacts Server 8.0.0.5.0 release onwards, a new parameter **appsrv.isweblogic** is created inside the statefile. Therefore, when Contacts Server is deployed on WebLogic Server, the **appsrv.isweblogic** parameter is set to **true**. If the **appsrv.isweblogic** parameter is set to **false** or the absence of this parameter indicates that Contacts Server is deployed on GlassFish Server.

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

init-config -s -f *path_to_statefile_saved_during_freshconfig_usingWeblogic*

Next Steps

After configuring Contacts Server, continue with the following chapters:

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

6

Configuring Contacts Server With Multiple Hosts

This chapter describes how to configure Oracle Communications Contacts Server with multiple hosts.

About Installing and Configuring Multiple Contacts Server Databases

A standard Contacts Server installation consists of a default back-end database that contains contact data. Perform the procedures in this section to add additional back-end data bases to your initial deployment.

In the case of multiple Contacts Server front ends, configure each to use the same initial default database back end. Then, you add additional back ends to each front end.

Each back-end database must have its own document store. When you use 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 configuration procedure follows these steps:

1. (Optional) If you have not done so already, install and configure the new database. Then prepare the new database by running either the **config-mysql** or **config-oracle** script.
2. Gather the database host names, ports, and other database names.
3. Install all front-end servers, if not already done.
4. Configure all front-end servers by running 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 about your back-end server is used for this step. See "Renaming the Default Contacts Server Database" if you have already run the **init-config** script, and you want to rename the default back-end server.
5. On each front-end server, run the **config-backend** script.

   Note: If you use WebLogic Server, you cannot run this script. See "Installing and Configuring Multiple Contacts Server Back-End Hosts for WebLogic Server Manually" for more information.
6. On each front-end server, enable connection pool validation.

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

7. On each front-end server, restart the application server.

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

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

1. Install the database software on each back-end host. Choose one of the following:
   - Installing and Configuring MySQL Server
   - Installing and Creating an Oracle Database Instance for Contacts Server
2. If you want to create additional MySQL Server databases, do the following:

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

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

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

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

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

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

   This script creates a JDBC connection pool and a JDBC resource on the GlassFish Server, and a back-end configuration using the user-provided back-end identifier.

   ```
   ContactsServer_home/sbin/config-backend
   ```

   Follow the prompts and enter the database and document store information.

   - If your deployment is using MySQL Server, make sure the value you enter for **Contacts db name on remote server** is the one that you used for the **config-mysql -c** command.
   - If your deployment is using Oracle Database, make sure the value you enter for **Contacts 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 using the user-provided back-end identifier.

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

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

   If needed, provision accounts for a multiple back-end deployment. See "Provisioning Accounts in a Multiple Back-End Deployment" for more information.

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

Renaming the Default Contacts Server Database

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

If you need to change this JDBC resource name, to match other naming conventions, follow these steps.

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

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

   ```
   store.dav.jdbc-backend-id.backendid=jdbc-backend-id
   store.dav.jdbc-backend-id.jndiname=jdbc/jdbc-backend-id
   ```

   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 **nabStore** attribute values.

> **Note:** After your Contacts Server deployment is up and running, do not change the user back-end ID as defined by the **nabStore** attribute. Otherwise, data in the original database becomes orphaned and inconsistency errors result for users.

Provisioning Accounts in a Multiple Back-End Deployment

For accounts to know which back-end host they should connect to, you must provision accounts with the **nabStore** attribute. The **nabStore** attribute indicates the back-end host that stores a user's data if the deployment is configured for multiple back ends. For more information, see the topic on Contacts Server LDAP attributes in *Communications Suite Schema Reference*.

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

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

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

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

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

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

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

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

3. To create the Oracle database user and schema, see the following:
   - Preparing Oracle Database 11g Release 2 or Oracle Database 12c Not Pluggable (non-CDB)
   - Preparing Oracle Database 12c Container Database
4. Collect the following details specific to your new database backend setup.
   - *Frontend*: host1.us.oracle.com: Contacts Server deployed on WebLogic Server exists on this host.
   - *Database host*: host2.us.oracle.com: Existing backend database in production. Currently, frontend Contacts Server on host1 is configured to the backend database.
   - *New*: Additional database to be added: host3.us.oracle.com: Required database (MySQL database or Oracle database) is setup on this machine. This is associated with identifier **backend2**.

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

   If you use MySQL database:

   - Remote database server host name= *host3.us.oracle.com*
   - Remote database server port= *3306*
   - Contacts database name on remote server=*dav*1
   - Contacts database user name=*mysql*
   - Contacts database user password=*mysql*

- Backend identifier for the remote database=*backend2*
- Document store directory (leave blank if store is remote)= */var/opt/sun/comms/nabserver/db/backend2*
- Document store host (leave blank if store is local)
- Document store port (leave blank if store is local)
- Application server admin user password=*adminpass* (Front-end WebLogic Server Admin User password)

If you use Oracle database:

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

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

- Document store host

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

- Document store port

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

- Application server administrator user password= *adminpass*

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

---

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

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

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

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

   ---

6. Log in to WebLogic Server Administration Console on host1.us.oracle.com. For example, https://host1.us.oracle.com:7002/console
7. Click **Lock & Edit**.
8. Click your domain directory. For example, domain1.
9. Navigate to **Services** and then **Data Sources**.
10. Under the **Configuration** tab, click **New**.
11. Select **Generic Data Source**.
12. Enter the following details:
    - name: backend2
    - scope: global
    - JNDI Name: jdbc/backend2
    - If your database type is MySQL, select Database type: MySQL
        a. Click **Next**.
        b. Database Driver: Select MySQL's Driver (Type 4) Versions: using com.mysql.jdbc.jdbc2.optional.MySqlDataSource
        c. Click **Next**.

           You can leave the default values shown under the Transaction options.
        d. Click **Next**.
        e. Database Name: dav1
        f. Hostname=host3.us.oracle.com
        g. port = 3306
        h. Database User Name=mysql
        i. password=mysql
        j. confirm password=mysql
        k. Click **Next**.
        l. Ensure that the following is shown in URL=jdbc:mysql://host3.us.oracle.com:3306/dav1
        m. Under Properties, (properties to pass to the JDBC driver when creating database connections), you can have the following: user=mysql, databaseName=dav1, characterEncoding=UTF8
        n. Test table Name: SQL SELECT 1
        o. Click **Test Configuration**.

           The result must be **Connection test succeeded**.
        p. Click **Next**.
    - If your database type is Oracle, select Database type: Oracle.
        a. Click **Next**.
        b. Database Driver: Select ***Oracle's Driver (Thin) for Service connections; Versions:Any**.
        c. Click **Next**.

d. You can leave the default values shown under Transaction options.

e. Click **Next**.

f. Database Name: pdb2.in.oracle.com

g. Hostname=host3.us.oracle.com

h. port=1521

i. Database User Name=nabadmin

j. password=password

k. confirm password=password

l. Click **Next**.

m. Ensure the URL is shown correctly as:
URL=jdbc:oracle:thin:@//host3.us.oracle.com:1521/pdb2.in.oracle.com

n. Test table Name: SQL SELECT 1 FROM DUAL

o. Click **Test Configuration**.

The result must be **Connection test succeeded**.

p. Click **Next**.

13. Select the target server.

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

---

14. Click **Finish**.

You can see the newly created backend2 in the list of **Data Sources** table.

15. Click **Activate Changes**.

16. Restart WebLogic Server.

Ensure that WebLogic Admin or Managed Server log file does not contain any errors.

17. Run **davadmin backend create** where a new resource is created. For example, *backend2*.

18. Log in to your front-end host where the Contacts Server is set up. For example, *host1.us.oracle.com*.

19. Navigate to the *ContactsServer_home*/sbin directory where *davadmin* CLI tool is residing. For example, **/opt/sun/comms/nabserver/sbin**/*davadmin*.

20. Run the *davadmin* command:

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

For example,

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

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

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

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

    store.dav.backend2.jndiname=jdbc/backend2

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

    store.dav.backend2.attachstorehost=

    store.dav.backend2.attachstoreport=8008

    store.dav.backend2.backendid=backend2

    store.dav.backend2.purgedelay=2592000

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

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

7

Configuring the Contacts Server Document Store

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

About Configuring the Document Store

The Contacts Server document store is used to store and retrieve ***large data***, such as photos and logos. You must set up one document store per configured Contacts Server back-end 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, Contacts Server accesses the file systems directly. When the document store is remote, Contacts Server accesses the documents by using either HTTP or HTTPS. If you use Oracle Database, you can configure it to contain both the contact 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 contacts data and the document store.

Configuring Oracle Database for Both Contact Data and Large Data

In the case of Oracle Database, you can configure it to contain both the contacts data and the data that would otherwise need 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 contact data and large data:

1. Set the **store.dav.*backend-name*.attachstorehost** parameter to a value of **dbdocstore**. For example:

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

   You must do this on all back ends.

2. Restart Contacts Server by restarting the application server.

Configuring a Local Document Store

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/nabserver/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 Contacts Server to use the document store directory:

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

5. Restart Contacts Server by restarting the application server.

Configuring a Remote Document Store

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/nabserver/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:

   ```
   ContactsServer_home/sbin/configure-as
   ```

5. Change the **store.dav.*backend*.attachstorehost** and **store.dav.*backend*.attachstoreport** parameters on each Contacts 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 has its own document store. See the **davadmin** command documentation for more information about how to change the **store.dav.*backend*.attachstorehost** and **store.dav.*backend*.attachstoreport** parameters.

6. Restart Contacts Server by restarting the application server.

If you need to start or stop the remote document store, the commands are:

```
ContactsServer_home/sbin/start-as
ContactsServer_home/sbin/stop-as
```

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

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

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

   ```
   ContactsServer_home/sbin/stop-as
   ```

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

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

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

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

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

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

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

Configuring Remote Document Store Authentication

You must configure password authentication of the connection between the remote document store server (which runs on the remote host where the store is located), and the document store client (which runs on every Contacts 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 Contacts Server host, use the **davadmin passfile create** command to set the document store password.

   For example:

   ```
   cd ContactsServer_home/sbin
   davadmin passfile create
   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 ContactsServer_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: <Type 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: <Type 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/nabserver/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 ContactsServer_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 need to 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 Contacts Server host and the document store. Configuring SSL between Contacts Server and the document store consists of the following high-level steps:

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

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

Configuring the Document Store for High Availability

You can configure Contacts 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 fails over to the next document store host on the list.
3. Keep the data on the document stores synchronized at all times.

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 Contacts Server hosts. See the topic on document store configuration parameters in ***Contacts Server System Administrator's Guide*** for more information about the **ashttpd.properties** file.

To migrate the document store:

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

   ```
   DocumentServer_home/sbin/stop-as
   ```
3. Manually move the existing documents from the old path to the newly configured 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 ***Contacts 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:

   ```
   DocumentServer_home/sbin/start-as
   ```

6. Restart Contacts Server by restarting the application server.

8

Contacts Server Post-Installation Tasks

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

Many Contacts Server post-installation tasks involve configuring system security. For security-related tasks, such as configuring SSL, see ***Contacts Server Security Guide***.

Changing the User Unique Identifier

Contacts 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 contacts database. As a result, recreated users cannot access their existing contacts.

To change the unique identifier:

Run the **davadmin config modify** 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 ***Contacts 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.

Contacts 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 Contacts 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 Contacts Server to scan attachments, such as photos, for viruses. Contacts 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 Contacts Server for virus scanning, see the topic on configuring virus scanning in ***Calendar Server System Administrator's Guide***.

Though this documentation is written for Calendar Server, it also applies to Contacts Server. The only exception is that Contacts Server does not have an iSchedule database.

Configuring Directory VLV Browsing for Contacts Server

This section describes how to set up a Virtual List View (VLV) browsing index for Oracle Directory Server Enterprise Edition 11.1.1.5.0 using Contacts Server. Directory Server VLV browsing indexes are used by Contacts Server to enable pagination support in the Corporate Address Book from the RESTful protocol. For more information, see the topic on managing browsing indexes in ***Oracle Directory Server Enterprise Edition Administration Guide*** at:

http://docs.oracle.com/cd/E20295_01/html/821-1220/bcaug.html

Using VLV in Contacts Server

By default, Contacts Server enables the Corporate Directory feature by using the deployment's user/group LDAP pool with the following configuration parameters:

- **store.corpdir.enablecorpdir=true**
- **store.corpdir.defaultcorpdirectoryurl=ldap://ugldap/??sub?(objectclass=*)**

If necessary, tailor the **objectclass** to your site's needs, for example: **objectclass=inetorgperson**.

However, VLV is not enabled by default. To use VLV, you must perform the following steps:

- Define a browsing index and generate the browsing index in Directory Server, as described in "Creating a VLV Browsing Index."
- Use an LDAP URL syntax extension to enable the use of VLV in the Contacts Server **store.corpdir.defaultcorpdirectoryurl** configuration parameter.

The general LDAP URL syntax is in the form l**dap://***ldap_pool_name*/*basedn***?attributes?scope?filter?extensions**. The available private extensions are:

- **vlv**
- **sort=***sort_on_attribute*
- **displayname=***friendly_display_name*

The **displayname** extension is optional. If it is missing, then the value "Corporate Directory" is used. Thus, to enable VLV in the default corporate directory, you set the **store.corpdir.defaultcorpdirectoryurl** parameter as follows:

```
store.corpdir.defaultcorpdirectoryurl =
ldap://ugldap/??sub?(objectclass=inetorgperson)?vlv,sort=cn,displayname="Default
Corporate Directory"
```

This value enables VLV, uses **cn** as the sorting attribute, and causes the **displayname** to appear in the output of listing all public address books:

```
>> Request <<
GET /rest/?booktype=public HTTP/1.1
Host: siroe.com
Content-Length: 0
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==

>> Response <<
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: xxxx
{
    "restversion": "1.0",
    "baseuri": "https://siroe.com",
    "homeuri": "/rest/home/johndoe/",
    "addressbook": [{
        "uri": "/rest/directory/default/",
        "displayname": "Default Corporate Directory",
        "type": "public",
        "myrights": "r"
    }],
    "totalresults": 1
}
```

Creating a VLV Browsing Index

You must perform the following steps on every Directory Server instance that Contacts Server uses. Directory Server index configuration settings are not replicated.

Creating a VLV browsing index involves the following tasks:

- Defining the VLV Browsing Index
- Generating the VLV Browsing Index

Defining the VLV Browsing Index

To define the VLV browsing index:

1. The following LDAP modifications add the required Directory Server settings for the VLV index, which enables you to then create the index in "Generating the VLV Browsing Index".

```
/opt/sun/dsee7/dsrk/bin/ldapmodify -h directory_server_hostname -p directory_
server_port -D "cn=Directory Manager"
dn: cn=Browsing organization,cn=database_backend,cn=ldbm
database,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: vlvSearch
cn: Browsing organization
vlvbase: organization_base
vlvscope: 2
vlvfilter: vlv_search_filter
aci: (targetattr="*")(version 3.0; acl "VLV for Anonymous";
 allow (read,search,compare) userdn="ldap:///anyone";)

dn: cn=Sort by vlv_sort_attribute,cn=Browsing organization,cn=database_
backend,cn=ldbm database,cn=plugins,cn=config
changetype: add
```

```
objectClass: top
objectClass: vlvIndex
cn: Sort by vlv_sort_attribute
vlvSort: vlv_sort_attribute
```

2. To determine the ***database_backend*** setting, to refer to the ***ds_instance_path*/config/dse.ldif** file and search for **nsslapd-suffix:** ***user_group_base***. The following example shows the ***database_backend*** as **isp** for the ***user_group_base*** of **o=isp**.

```
dn: cn=isp,cn=ldbm database,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
objectClass: nsBackendInstance
cn: isp
creatorsName: cn=directory manager
modifiersName: cn=directory manager
entrydn: cn=isp,cn=ldbm database,cn=plugins,cn=config
numSubordinates: 4
nsslapd-suffix: o=isp
nsslapd-cachesize: -1
nsslapd-cachememsize: 10485760
nsslapd-readonly: off
nsslapd-require-index: off
nsslapd-directory: /var/opt/SUNWdsee/dsins1/db/isp
```

3. The following **ldapmodify** command uses the preceding value for ***database_backend***:

```
/opt/sun/dsee7/dsrk/bin/ldapmodify -h directory.siroe.com -p 389 -D
"cn=Directory Manager"
dn: cn=Browsing siroe.com,cn=isp,cn=ldbm database,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: vlvSearch
cn: Browsing siroe.com
vlvbase: o=siroe.com,o=isp
vlvscope: 2
vlvfilter: (objectclass=inetorgperson)
aci: (targetattr="*")(version 3.0; acl "VLV for Anonymous";
 allow (read,search,compare) userdn="ldap:///anyone";)

dn: cn=Sort by cn,cn=Browsing siroe.com,cn=isp,cn=ldbm
database,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: vlvIndex
cn: Sort by cn
vlvSort: cn
```

Generating the VLV Browsing Index

To propagate the VLV index with data, you must first stop the Directory Server.

1. Change to the ***ds_install*/bin** directory and stop the Directory Service.

```
cd /opt/sun/dsee7/bin
dsadm stop ds_instance_path
```

2. Generate the index.

```
dsadm reindex -l -t "Sort by vlv_sort_attribute" ds_instance_path
```

```
"organization_base"
```

3. Start the service.

```
dsadm start ds_instance_path
```

The following commands show how to generate a VLV index using the preceding example settings:

```
cd /opt/sun/dsee7/bin
dsadm stop /var/opt/sun/dsee7/dsins1/
dsadm reindex -l -t "Sort by cn" /var/opt/sun/dsee7/dsins1/ "o=siroe.com,o=isp"
dsadm start /var/opt/sun/dsee7/dsins1/
```

Verifying the VLV Index

To verify that the VLV index is created and in use, run the **ldapsearch** command. Because Contacts Server uses an internal administrative user to proxy as a regular user, the following example does the same.

```
/opt/sun/dsee7/dsrk/bin/ldapsearch -h directory.siroe.com -p 389 -D \
"uid=nab-admin-contactsserver.siroe.com-20140314084929Z,ou=People,o=siroe.com, \
o=isp" -Y "dn:uid=johndoe,ou=People,o=siroe.com,o=isp" -w - -b o=siroe.com,o=isp \
-G 0:4:1:0 -S cn -x "(objectclass=inetorgperson)" dn cn
```

Log entries similar to the following appear in the Directory Server access log for the preceding search, indicating that the VLV is set up properly.

```
[26/Mar/2014:08:11:53 +0000] conn=14037 op=1 msgId=2 - SRCH
base="o=siroe.com,o=isp" scope=2 filter="(objectClass=inetorgperson)" attrs="dn
cn"
[26/Mar/2014:08:11:53 +0000] conn=14037 op=1 msgId=2 -  SORT cn
[26/Mar/2014:08:11:53 +0000] conn=14037 op=1 msgId=2 -  VLV 0:4:0:0 1:212689 (0)
[26/Mar/2014:08:11:53 +0000] conn=14037 op=1 msgId=2 - RESULT err=0 tag=101
nentries=5 etime=0
```

Configuring Directory Server Telephone Indexes for Contacts Server

This section describes how to set up indexes for Oracle Directory Server Enterprise Edition using Contacts Server for the following telephone attributes:

- **telephoneNumber**
- **facsimileTelephoneNumber**
- **homePhone**
- **mobile**
- **pager**

By default, the **comm_dssetup** script only creates an index on **telephoneNumber**.

Creating the Telephone Indexes for Contacts Server

You must perform the following steps on every Directory Server instance that Contacts Server uses. Directory Server index configuration settings are not replicated.

1. Copy the **index-odsee.sh** script from the ***ContactsServer_home*/sbin** directory to the Directory Server host.
2. On the Directory Server host, change to the directory in which you copied the script.

3. Run the **index-odsee.sh** script.

```
index-odsee.sh -B dsbinpath -D binddn -j passwdfile -h dshost -p dsport -s
suffix
```

where:

-B *dsbinpath*: Specifies the path to the Directory Server binary location

-D *binddn*: Specifies the bind dn

-j *passwdfile*: Reads the bind password from a file for simple authentication

-h *dshost*: Specifies the Directory Server host name

-p *dsport*: Specifies the Directory Server port number

-s *suffix*: Specifies the directory suffix where indexes are to be created

9

Verifying the Contacts Server Installation

This chapter describes how to verify that Oracle Communications Contacts Server is installed correctly.

Verifying the Contacts Server Installation

To verify the Contacts Server configuration, you configure a client to connect to the server.

Prerequisites for Configuring Contacts Server Clients

- Users need to be provisioned in Directory Server. For more information, see Directory Server and schema information in ***Calendar Server Concepts***.
- Obtain the following information on your Contacts Server deployment:
  - Application server host name and port number where Contacts Server is installed
  - User identifier (email address or ***uid*@*domain***)
  - If SSL is used

Configuring CardDAV Clients

This information describes how to configure Android, Apple, and Lightning clients to communicate with Contacts Server.

Contacts Server Addresses to Use for Configuring Clients

Use the following server addresses when configuring Contacts Server clients:

- If the deploy URI is **/**, use:

  `http://example.com:`*`port`*`/`

- If the deploy URI is **/nabserver**, use:

  `http://example.com:`*`port`*`/nabserver`

For iOS and Mac OS clients, you can use any of the following server addresses:

`http://example.com:`*`port`*`/`
or
`http://example.com:`*`port`*`/dav/principals/`*`user`*`@`*`domain`*`/`
or
`http://example.com:`*`port`*`/dav/principals/`*`user`*`@`*`domain`*`/addressbook/`

If you deployed Contacts Server on **/** as the context root, with the default HTTP and HTTPS ports (80 and 443 respectively), then clients construct the URI on their own and you do not have to enter a value.

Samsung and Thunderbird clients require that you enter the entire server address, such as the following:

`http://example.com:`*`port`*`/dav/home/`*`user@domain`*`/addressbook/`

Configuring a Card-DAV Sync Client

These steps apply to Card-Dav Sync, at least version 0.4.5.

1. Launch Card-DAV Sync.
2. Select **CardDAV** for Add account.
3. Enter the server, user name, and password.
4. If SSL is not enabled for Contacts Server, deselect the **Use SSL** check box.
5. Press **Next**.
6. If desired, change the account name.
7. Deselect the **Sync from server to phone only** option.
8. Press **Finish**, and press **OK** when the warning message appears.

Configuring an Apple Mac OS Client

These steps apply to MAC OS X 10.9 client, at least version 8.0.

1. Launch Contacts.
2. Select **Preferences** from the Contacts menu.
3. To add a new account, click the **Add (+)** button.
4. Select **Other contacts account** to add the contacts account and press **Continue**.
5. Enter the user name, password and server address. Ensure that **cardDAV** is selected then press **Create**.

Configuring an Apple iOS Client

These steps apply to iOS Client, at least version 7.0.2.

1. Launch Settings.
2. Navigate to the Mail, Contacts, and Calendar settings menu.
3. Select **Add Account**.
4. Select **Other**.
5. Select **Add CardDAV Account** under the Contacts section.
6. Enter the server, user name, and password, then press **Next**.

Configuring a Thunderbird Client

These steps apply to Thunderbird client, at least version 24.0.2. Before starting, ensure that you have installed the Inverse SOGo Connector.

1. Launch ThunderBird.
2. Navigate to Address Book.

3. From the **File** menu, select **New**.
4. Click **Remote Address Book**.
5. Enter the name and server details.
6. Ensure that **Read only** is unchecked then press **ok**.
7. Close then relaunch Thunderbird.
8. When prompted by Thunderbird, enter the user name and password.

Importing and Exporting Contacts

The **davadmin contact** command enables you to import and export address book contacts. See the topic on the **davadmin contact** command in ***Contacts Server System Administrator's Guide*** for more information.

10

Upgrading Contacts Server

This chapter explains how to upgrade your existing system to the latest release of Oracle Communications Contacts Server.

Note: If you are migrating Contacts Server deployment from GlassFish Server to WebLogic Server, perform the following steps:

1. Ensure to upgrade your existing Contacts Server deployed on GlassFish Server to Contacts Server version 8.x (the recommended version is, 8.0.0.4.0).
2. Set up WebLogic Server on a new server with Contacts Server version 8.0.0.5.0 or above which has WebLogic container support.
3. Ensure to configure your Contacts Server on WebLogic Server deployment with your required backend server accordingly when migration is completed.

About Upgrading Contacts 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 Contacts 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 Contacts Server versions. Contacts 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 Contacts Server Database Upgrades

Each Contacts 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 Contacts Server database schema or structure upgrade is required, allow additional time for the overall Contacts Server upgrade, as the database upgrade can take a while, especially if the database is large.

Topics in this section:

- About Upgrading the Database Schema
- Preupgrade Functions

Database Schema and Structure Version

Each database schema and structure has a Contacts Server back-end version number recorded in the database. Each Contacts 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 Contacts Server version, the previous Contacts Server version does not recognize the newly upgraded database. However, not all Contacts Server versions involve a database schema upgrade. When this is the case, you can revert to a previous Contacts Server version without invalidating the databases. Use Table 10–1, " Database Schema Version Changes for Contacts Server" to understand which Contacts Server versions use the same database schema.

Contacts Server Start Up and Automatic Database Upgrade

When Contacts Server starts up in the application 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, Contacts Server checks the database schema version. Contacts 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 Contacts Server or application server while the database is being upgraded.

Contacts 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 Contacts 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 Contacts Server upgrade, you must make a backup before starting the upgrade process. Once a database is upgraded, it no longer works with a previous Contacts Server version. Use Table 10–1 to understand the Contacts Server database changes that have occurred.

Table 10–1 Database Schema Version Changes for Contacts Server

Contacts Server Release	Database Change	Comments
8.0.0.3	Automatic upgrade to version 10.	Fairly intensive, depending on the size of existing data.

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 10–1, " Database Schema Version Changes for Contacts Server" 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 10–2 describes these two commands.

Table 10–2 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 Contacts Server instance, which upgrades the database. Use this command to upgrade multiple back-end databases in parallel instead of one at a time, as happens during the normal Contacts Server upgrade process. For example, if your deployment consists of four back-end databases, instead of going through the normal upgrade process where upgrading a front-end host then initiates the back-end database upgrade, one at a time, you can upgrade the front-end hosts then upgrade all back-end databases in parallel by using the **schema_fullupgrade** option. After running the **schema_fullupgrade** command and the database schema upgrade completes, you cannot revert to the previous schema. In addition, Contacts Server front-end hosts that run a prior software version are not able to communicate with the database that has had its schema upgraded.	The **schema_preupgrade** command enables you to perform online database DDL changes, as well as a partial offline upgrade, as long as the release contains database schema updates. (Not all releases update the database schema, so **schema_preupgrade** might not apply.) For example, if your deployment contains a large amount of data, you can save time by pre-upgrading the database schema in advance of the formal Contacts Server software upgrade. The **schema_preupgrade** command does not change the database schema version, and therefore, Contacts Server 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 10–3 for a list of functions by release. For example, if a Contacts 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 Contacts 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 Contacts 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 "Contacts Server Command-Line Utilities" in ***Contacts 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 10–3 describes the available functions by release.

Table 10–3 schema_preupgrade Functions

Database Version	Function	Description
Version 10	**v10cardprop**	Upgrades the **CardProp** table.

Upgrading to Contacts Server 8.0.0.x

This section describes how to upgrade to Contacts Server 8.0.0.x. Depending on your current Contacts Server version, some steps are not required.

Prerequisites for Upgrading Contacts Server

1. Download the Contacts Server software and Directory Server setup script, **comm_dssetup** (if a newer version is required), either as part of a media pack, or as a patch.

- Download the media pack from the Oracle software delivery website.
- Download a Contacts Server patch or Directory Server setup patch from My Oracle Support. Contacts Server 8.0.0.x currently requires at least version 6.4-26.01 of **comm_dssetup**.

2. Place the Contacts Server software on the appropriate Contacts Server hosts, and the Directory Server setup script software on the appropriate Directory Server hosts.

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

Backing Up the Contacts Server Database

The purpose of backing up the database is in case you must restore the previous Contacts 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 ***Contacts 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 Contacts Server upgrade process, if pre-upgrade functions exist for the release. (Not all Contacts 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 10–1, " Database Schema Version Changes for Contacts Server" to confirm if the Contacts Server version upgrades the database schema. See the topic on the **davadmin db schema_preupgrade** command in ***Contacts Server System Administrator's Guide*** for more information.

Preparing the Directory Server

Make sure you have run the required version of the **comm_dssetup.pl** script against your Directory Server hosts.

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 "commpkg Reference".

Running commpkg upgrade

This section assumes that you have previously put the Contacts Server software onto each Contacts Server front-end host and remote document store host, if applicable. If

instead you are performing an upgrade of the Contacts 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 Contacts Server front-end hosts, run **commpkg upgrade** and select the Contacts Server component.

   For more information, see "upgrade Verb Syntax".

   If the upgrade detects a problem during the pre-upgrade phase, correct the problem and re-run the upgrade.

2. If you have remote document stores, continue with the next section.

Upgrading the Remote Document Store

When upgrading Contacts 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 Contacts Server software zip file or Contacts 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 ContactsServer_home/sbin
   stop-as
   ```

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

   ```
   commpkg upgrade
   ```

   c. Configure the document store and set the password.

   ```
   cd ContactsServer_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/nabserver/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 ContactsServer_home/sbin
start-as
```

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

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

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

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

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

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

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

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

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

   b. Restart the application server.

Upgrading a Non-SSL Document Store to SSL

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 ContactServer_home/sbin
   stop-as
   ```

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

   ```
   commpkg upgrade
   ```

2. Create a self-signed certificate.

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

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

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

4. Configure the document store.

   ```
   cd ContactsServer_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/nabserver/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 Contacts Server front-end host after you have upgraded the Contacts Server software.
8. Import the certificate (the certificate that you copied from the document store host) into the TrustStore on the application server host where you have deployed Contacts Server.

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

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

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

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

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

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

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

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

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

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

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

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

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

12. Restart the application server.

Performing the Contacts Server Configuration

To perform the configuration:

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

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 Contacts Server to set the presence triggers for existing future events.

To set presence triggers after upgrading:

```
davadmin acount 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 Contacts Server functionality.

11

Uninstalling Contacts Server

This chapter describes how to uninstall Oracle Communications Contacts Server.

Uninstalling Contacts Server

The **commpkg uninstall** command enables you to uninstall Communications Suite products, such as Contacts 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 Contacts Server:

1. Log in as **root**.
2. Change to the ***InstallRoot*/CommsInstaller/bin/** directory.
3. Run the **commpkg uninstall** command.
4. Choose Contacts Server from the list of installed Communications Suite components.
5. When prompted, enter **Yes** to continue.

12

Installing Patches

This chapter describes how to install patches on Oracle Communications Contacts Server.

About Patching Contacts Server

Contacts 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 Contacts 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 Contacts 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 and Linux 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 Contacts Server:

1. Back up your Contacts 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 the application server from servicing incoming requests.

   If you use GlassFish Server, see "Disabling GlassFish Server Incoming Connections During Patch Application" for more information.

   If you use WebLogic Server, stop the Managed Server or target server on which the application is deployed.

3. Apply the patch by running the following command.

   ```
   commpkg upgrade
   ```

4. Run the Contacts Server **init-config** script to enter the current deployment configuration values.

5. Run the Contacts 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 the application 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 Contacts Server.

Disabling GlassFish Server Incoming Connections Overview

GlassFish Server needs to be up and running when you apply a patch to Contacts 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 Contacts 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 Contacts 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

commpkg Reference

This appendix provides information about the Oracle Communications Contacts Server **commpkg** command.

Overview of the commpkg Command

The **commpkg**, command, also referred to as the installer, comprises several commands (verbs) that enable you to install, uninstall, and upgrade Contacts Server software and its shared components. The **commpkg** command is installed in the directory in which you extract the product software.

Syntax

```
commpkg [general_options] verb [verb_options]
```

Table A–1 describes the **commpkg** command general options.

Table A–1 commpkg Command General Options

Option	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--OSversionOverride	Overrides the operating-system version check.
--fixEntsys [**y** | **n**]	Fixes an invalid Sun Java Enterprise System (Java ES) **entsys symlink**, making the link point to the latest Java version upgraded by **commpkg**. The Java ES symlink is located in **/usr/jdk/entsys-j2se**. Choose **--fixEntsys y** to fix the Java ES symlink to the Java files. If you do not specify this switch, **commpkg** prompts you if the symlink is invalid. However, in silent mode, the default is not to fix the symlink (the equivalent of using a value of n). To fix the symlink in silent mode, type **commpkg install --fixEntsys y --silent** ***INPUTFILE*** on the command-line.

Table A–2 describes the **commpkg** command verbs.

Table A–2 commpkg Command Verbs

Verb	Description
install	Performs software installation.
uninstall	Uninstalls software but does not remove OS patches or shared components installed by **commpkg install**.

Table A–2 (Cont.) commpkg Command Verbs

Verb	Description
info	Displays product information on the paths (also known as ***installroots***) where Contacts Server is installed, and the products that are installed in those paths.
upgrade	Performs software upgrade.
verify	Verifies installed product.

install Verb Syntax

commpkg install [*install_options*] [*ALTROOT*|*name*]

> **Tip:** Installing Only Shared Components: To install just the product's shared components, launch the installer then prefix your product selection with a tilde (~). You can type multiple selections by using a comma to separate the entries.

Table A–3 describes the **commpkg install** verb options.

Table A–3 commpkg install Options

commpkg install Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Does not apply operating system patches during product installation.
--excludeSC	Does not install, upgrade, or patch any shared components.
ALTROOT | ***name***	Use this option to install multiple instances of the product on the same host or Oracle Solaris zone. You can also use this option to perform a side-by-side upgrade of the product. This option is available on Solaris only. Specifies an alternate root directory for a multi-instance installation. The ***InstallRoot*** (the top-level installation directory for all products and shared components) is the alternate root. If you specify a ***name***, it will be a friendly name associated with the ***ALTROOT*** that is registered in the software list. If you specify the ***name*** and it exists in the software list, the corresponding ***ALTROOT*** is used. If you also specify the **--installroot** option, it must correspond to the entry in the software list. If you specify name and it does not exist in the software list, it is added to the software list. Specifying any ***name*** other than "" implies an **ALTROOT**. A value for ***name*** of "" is reserved for the default root.
--installroot	Specify location of INSTALLROOT, the top level installation directory for all products and shared components. The top-level installation directory for individual products are subdirectories under INSTALLROOT. Default is **/opt/sun/comms**.
--distro ***path***	Specifies the ***path*** to packages or patches for the products. Default: Location of **commpkg** script

Table A–3 (Cont.) commpkg install Options

commpkg install Options	Description
--silent *INPUTFILE*	Runs a silent installation, taking the inputs from the *INPUTFILE* and the command-line arguments. The command-line arguments override entries in the *INPUTFILE*. Installation proceeds without interactive prompts. Use **--dry-run** to test a silent installation without actually installing the software. Specify **NONE** for *INPUTFILE* to run in silent mode without using an input file. When you specify **NONE**, the installation uses default values.
--dry-run or **-n**	Does not install software. Performs checks.
--upgradeSC [**y**|**n**]	Upgrades or does not upgrade shared components as required. If this option is not specified, you are prompted for each shared component that must be upgraded by using package removal and installation. Default: **n** **Caution:** Upgrading shared components by using package removal and installation is irreversible. However, if you do not upgrade required shared components, products might not work as designed. The **--excludeSC** flag has precedence over this flag.
--auditDistro	Audits the installation distribution to verify that the patches and packages matches the versions in the product files internal to the installer.
--pkgOverwrite	Overwrites the existing installation package. You might use this option when you are installing a shared component in a global zone where either the shared component does not exist in a global zone, or the shared component exists in the whole root zone. The default is not to override the existing package. In general, shared components should be managed in the global zone.
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.
--skipOSLevelCheck	(Solaris only) The installer always checks that the operating system is at a certain minimum patch level. Using this option skips the check.

uninstall Verb Syntax

commpkg uninstall [*verb_options*] [*ALTROOT*|*name*]

Table A–4 describes the **commpkg uninstall** verb options.

Note: A fast way to uninstall a product that was installed in an alternate root is to simply remove the entire alternate root directory.

Table A–4 commpkg uninstall Options

commpkg install Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--silent *INPUTFILE*	Runs a silent uninstall, taking the inputs from the ***INPUTFILE*** and the command-line arguments. The command-line arguments override entries in the ***INPUTFILE***. Uninstall proceeds without interactive prompts. Use **--dry-run** to test a silent installation without actually installing the software.
--dry-run or **-n**	Does not install software. Performs checks.
ALTROOT | *name*	Use this option to uninstall multiple instances of the product on the same host or Oracle Solaris zone. You can also use this option to perform a side-by-side upgrade of the product. This option is available on Solaris only. Specifies an alternate root directory for a multi-instance uninstallation. The ***InstallRoot*** (the top-level installation directory for all products and shared components) is the alternate root. If you specify a ***name***, it will be a friendly name associated with the ***ALTROOT*** that is registered in the software list. If you specify the ***name*** and it exists in the software list, the corresponding ***ALTROOT*** is used. If you also specify the **--installroot** option, it must correspond to the entry in the software list. If you specify name and it does not exist in the software list, it is added to the software list. Specifying any ***name*** other than "" implies an **ALTROOT**. A value for ***name*** of "" is reserved for the default root.

upgrade Verb Syntax

commpkg upgrade [*verb_options*] [*ALTROOT*|*name*]

Table A–5 describes the **commpkg upgrade** verb options.

Table A–5 commpkg upgrade Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Does not apply operating system patches during product upgrade.
--excludeSC	Does not install, upgrade, or patch any shared components.

Table A–5 (Cont.) commpkg upgrade Options

Options	Description
ALTROOT | *name*	This option is available on Solaris only. Specifies an alternate root directory during a multiple host installation. The ***InstallRoot*** (the top-level installation directory for all products and shared components) is the alternate root. If you specify a ***name***, it is an "alias" associated with the alternate root that is registered in the software list. You can use this option to upgrade multiple product instances on the same host or Solaris zone. Additionally, you can use this option to perform a side-by-side product upgrade.
--distro *path*	Specifies the ***path*** to packages and patches for the products. Default path: Location of the **commpkg** command.
--silent *INPUTFILE*	Runs a silent upgrade, taking the inputs from the ***INPUTFILE*** and the command-line arguments. The command-line arguments override entries in the ***INPUTFILE***. Upgrade proceeds without interactive prompts. Use **--dry-run** to test a silent upgrade without actually installing the software. Specify **NONE** for ***INPUTFILE*** to run in silent mode without using an input file. When you specify **NONE**, the upgrade uses default values.
--dry-run or **-n**	Does not upgrade software but performs checks. This option creates a silent upgrade file in the **/tmp** directory.
--upgradeSC [**y**|**n**]	Indicates whether to upgrade shared components as required. If this option is not specified, you are prompted for each shared component that must be upgraded by the package uninstall/install. Default: **n** **Caution:** Upgrading shared components is irreversible. However, if you do not upgrade required shared components, products might not work as designed. The **--excludeSC** flag has precedence over this flag.
--pkgOverwrite	This option is only for Solaris systems. Overwrites the existing installation package. You might use this option when you are installing a shared component in a global zone where either the shared component does not exist in a global zone, or the shared component exists in the whole root zone. The default is not to override the existing package. In general, shared components should be managed in the global zone.
--components *comp1 comp2...*	Specifies products to be upgraded. Separate each component product with a space. (Thus, the list is a space-delimited set.) To specify each component product, use the mnemonic associated with that product. To display a list of the mnemonics for all the component products, use the **commpkg info --listpackages** command.
--usePkgUpgrade	If the upgrade can be performed by using a patch or an in-place package upgrade, this option uses the in-place package upgrade. The default is to use a patch to upgrade, if possible. **Note:** Patches are used only on Solaris.

verify Verb Syntax

commpkg verify [*verb_options*] [*ALTROOT*|*name*]

Tip: When verifying the package installation in an alternate root, be aware that Contacts Server uses the operating system components installed in the default root. Some products might also use shared components in the default root. Thus, verify the package installation in the default root as well.

Table A–6 describes the **commpkg verify** verb options:

Table A–6 commpkg verify Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--excludeOS	Do not verify operating system components.
--excludeSC	Do not verify shared components.
--components ***comp1 comp2...***	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.
ALTROOT | ***name***	Use this option to verify multiple instances of the product on the same host or Solaris zone. This option is available on Solaris only. Specify ***ALTROOT*** or ***name*** for an alternate root directory on which to perform the package verification.

info Verb Syntax

commpkg info [*verb_options*] [*ALTROOT*|*name*]

Table A–7 describes the **commpkg info** verb options.

Table A–7 commpkg info Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the installer version.
--clean	Removes entries in the software list. If ***ALTROOT*** | ***name*** is specified, the option removes the entry from the software list. If no ***ALTROOT*** | ***name*** is specified, the option removes all entries from the software list.
--listPackages	Lists the packages that comprise Contacts Server, shared components, and operating system auxiliary products. This option also displays the mnemonic for Contacts Server and components such as **comm_dssetup.pl**.
--verbose	Prints product information installed in the ***installroots***. To print information for a specific ***installroot***, run the following command: **commpkg info --verbose** ***installroot***

Table A–7 (Cont.) commpkg info Options

Options	Description
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you must escape the space between each mnemonic, that is, by adding double quotes around all the components.

About the Alternate Root

You can install multiple copies of the same product version on the same Solaris machine or Solaris zone by using the alternate root feature of the **commpkg install** command. For example, you might deploy a host with an installation in the default root directory, **/opt/sun/comms**, and a second, separate installation in the **/opt/sun/comms2** alternate root directory. The alternate root installation directory is the top-level directory underneath which the Contacts Server component product and shared components are installed in their respective directories.

Some possible uses for multiple installations include:

1. Performing a side-by-side upgrade.
2. Enabling an installation to be easily moved from one machine to another.

> **Note:** The alternate root feature is available only on Solaris. This feature is a "power user" feature. If you are interested in installing more than one instance of the same version of Contacts Server on the same physical host, another option is to use Solaris zones. For more information, see "Installing on Solaris OS Zones: Best Practices".

ALTROOT | name Syntax and Examples

You can use the optional ***ALTROOT | name*** option with the **commpkg install**, **commpkg upgrade**, **commpkg uninstall**, and **commpkg verify** commands. You use either ***ALTROOT*** or ***name***. The ***name*** acts as an alias for the alternate root installation path. The ***name*** is registered in an internal software list maintained by the installer. You can use ***name*** for the alternate root's path in commands that accept the alternate root. The distinction between the alternate root and name is that the alternate root always begins with a slash (/) whereas the name does not.

Syntax:

commpkg [**install**|**upgrade**|**unistall**|**verify**] [*ALTROOT*|*name*]

Example 1:

```
commpkg install /opt/sun/comms2
```

In this example, the path **/opt/sun/comms2** is the alternate root, which becomes the top-level directory underneath which Contacts Server software and shared components are installed.

Example 2:

```
commpkg install Comms2
```

In this example, **Comms2** is the name for the alternate root. During the installation process, the installer prompts you to type in the alternate root installation directory.

Example 3:

In this example, you avoid installing the shared components in the alternate root by using the **--excludeSC** option:

```
commpkg install --excludeSC /opt/sun/comms2
```

Example 4:

To install only the shared components, run the **commpkg install** command and select the product you want to install, but prepend a tilde (~).

For example, if you plan to install Contacts Server in the alternate root, you select ~1 during the default installation. This tells the installer to install the dependencies but not the product itself.

Notes on the ***ALTROOT*** | ***name*** command-line argument:

- Specifying a slash (/) as an alternate root is the same as specifying the default root installation directory. That is, specifying a slash is interpreted by the installer as having specified no alternate root.
- Likewise, specifying "" as an alternate root is interpreted as having specified no alternate root. (The "friendly name" for the default alternate root is "".)
- If you specify the **--installroot** option in addition to ***ALTROOT*** | ***name***, the two must match.
- Operating system patches are always installed into the default root (/). Some shared components are installed into the ***ALTROOT*** and some are installed into the default root (/).
- You can quickly uninstall an ***ALTROOT*** installation by using the **rm -r** command on the alternate root directory. The next time that you run the **commpkg info** command, the internal software list that maintains the alternate root information is updated.
- You can create as many alternate roots as you like. Running the **commpkg info** command reports on the various alternate roots.

Understanding the Difference Between ALTROOT and INSTALLROOT

The following concepts define an alternate root (ALTROOT):

- An alternate root directory is a Solaris feature that is used by the **commpkg** command to enable multiple product installations on the same host.
- The default alternate root is the standard root directory (/) and is always present.

The following concepts define an installation root (***InstallRoot***):

- An ***InstallRoot*** is the top-level umbrella installation path for Contacts Server.
- On the default alternate root (that is, /), you can specify an ***InstallRoot***.
- On an alternate root, the ***InstallRoot*** is the same as the alternate root.

Default Root

If you use the default root, the default ***InstallRoot*** is:

/opt/sun/comms/

Using Both Default Root and Alternate Root

Suppose you want to install one instance of Contacts Server in the **/opt/sun/mycompany/comms/** directory, and another instance of the same product in the **/opt/sun/mycompany/comms2/** directory. You would use the following commands:

For the default root:

```
commpkg install --installroot /opt/sun/mycompany/comms
```

For the alternate root:

```
commpkg install /opt/sun/mycompany/comms2/
```

Running Multiple Installations of the Same Product on One Host: Conflicting Ports

By default, after you initially configure the product on alternate roots, the ports used by the different product installations are the same and thus conflict with each other.

This is not a problem if you install multiple installations of the same product on the same host but only intend to have one instance running at one time. For example, you may perform a side-by-side upgrade scenario in which you plan to stop the old instance before you start the new instance.

However, you may plan to test the new instance while the old instance is still running (and supporting end users). In this scenario, the ports are used simultaneously, and you must take steps to resolve the port conflicts.

B

comm_dssetup.pl Reference

This appendix provides information about the Oracle Communications Contacts Server **comm_dssetup.pl** script. You must prepare your Oracle Directory Server Enterprise Edition (Directory Server) hosts by running the **comm_dssetup.pl** script before you install and configure Contacts Server.

About the comm_dssetup.pl Script

This section provides information you need to understand before running the **comm_dssetup.pl** script.

The **comm_dssetup.pl** script performs the following steps:

1. Prompts you for your deployment's Directory Server and schema information.

   For a list of the specific information this step requests, see "Information Needed to Run the comm_dssetup.pl Script".

2. Generates a shell script and LDIF file from the information that you supply that is used to modify the Directory Server LDAP.

3. Runs the generated shell script and modifies your Directory Server.

At the end of each step, the **comm_dssetup.pl** script prompts you to continue. No changes are made to the Directory Server LDAP until the last step.

Directory Server Considerations for the comm_dssetup.pl Script

When running the **comm_dssetup.pl** script, consider the following points.

- **comm_dssetup.pl** configures local Directory Server instances, and thus you must:
  - Install the **comm_dssetup.pl** script on every host on which a Directory Server instance resides.
  - Run the **comm_dssetup.pl** script on the same host as your Directory Server. The tool runs locally for a specific instance (specified by path of Directory Server or path of instance).
- You can run the **comm_dssetup.pl** script against any Directory Server instance on the local host. If you have multiple Directory Information Trees (DITs) on one host, you can maintain and update one installation of **comm_dssetup.pl**, and apply it to every Directory Server instance on the host.
- **comm_dssetup.pl** must configure every Directory Server instance for the same DIT. This assumes that:

  - A Directory Server has already been installed, configured, and is running before you launch the **comm_dssetup.pl** script.
  - When adding an additional Directory Server host (such as a replica), at a future date, you must run the **comm_dssetup.pl** script against it, too.

- If you have customized your Directory Server, the following considerations might apply:
  - If you have indexed some attributes, you might have to reindex those attributes after running the **comm_dssetup.pl** script.
  - If you have added other LDIF files (schema definitions), they should not be affected, so no action should be necessary. However, back up your custom schema definition files before running the **comm_dssetup.pl** script.

    The **comm_dssetup.pl** script backs up old schema files to the **/var/tmp/dssetup_timestamp/save** directory.
  - For all Directory Server customizations, including the first two just listed, stop the **comm_dssetup.pl** script after it generates the script and before it actually updates the LDAP directory. Then inspect the script to evaluate how its proposed actions affect your LDAP directory. Take whatever actions you think necessary to protect your customizations before running the script against your Directory Server.

Information Needed to Run the comm_dssetup.pl Script

Table B–1 describes the information about your deployment that you need to gather before running the **comm_dssetup.pl** script.

Table B–1 comm_dssetup.pl Information

Information Item Needed	Default Value
Directory Server root path name	The default depends on the Directory Server version detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Which instance of Directory Server to use? (if more than one)	The default depends on the Directory Server version detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Directory Manager Distinguished Name (DN)	"**cn=Directory Manager**"
Directory Manager's Password	NA
Directory Server being used for user/group data? (yes), or configuration data only? (no)	**yes**

Table B–1 (Cont.) comm_dssetup.pl Information

Information Item Needed	Default Value
User and group root suffix (if yes to previous question)	The default depends on what is detected. The **comm_dssetup.pl** script does attempt to heuristically determine the value.
Schema version? (pick one of the following): ■ 1 - Schema 1 ■ 1.5 - Schema 2 Compatibility Mode ■ 2 - Schema 2 Native Mode For more information on how to choose a schema, see "About the comm_dssetup.pl Script Schema Choices". If you have one version of the schema installed and want to upgrade to a higher level, refer to ***Sun Java System Communications Services 6 2005Q4 Schema Migration Guide*** before running the script.	**2** However, if you run **comm_dssetup.pl** again, it defaults to the value that you chose the previous time.
If you choose Schema 1 or 1.5, you need a DC tree. If the DC tree does not yet exist, the **comm_dssetup.pl** script creates only the root suffix node, its does not create the rest of the DC tree. You must create the rest of your DC tree yourself.	**o=internet** However, if you run **comm_dssetup.pl** again, it defaults to the value that you chose the previous time.

About the Directory Server Root Path Name and Instance

The **comm_dssetup.pl** script prompts you for both the Directory Server root path and the Directory Server instance. The script then combines these two items into an absolute path name to the Directory Server instance. For example, if your Directory Server instance resides under the **/var/opt/sun/directory/slapd-varrius** directory, then you specify **/var/opt/sun/directory** for the Directory Server root path and **slapd-varrius** for the Directory Server instance.

The reason for having two **comm_dssetup.pl** prompts to specify one absolute path is historical. Prior to Directory Server 6, Directory Server had the concept of a "server root" under which all Directory Server instances (and the Directory Server binaries) resided. After Directory Server 6, the concept of the "server root" was removed. Directory Server instances (and the Directory Server binaries) do not all have to reside under a single umbrella "server root" directory.

About the comm_dssetup.pl Script Schema Choices

Contacts Server supports the following schema choices:

- LDAP Schema 2 native mode

  Corresponds to **comm_dssetup.pl** script schema version choice 2. This is the default for a fresh installation.

- LDAP Schema 1

  Corresponds to the **comm_dssetup.pl** script schema version choice 1.

- LDAP Schema 2 compatibility mode

  Corresponds to **comm_dssetup.pl** script schema version choice 1.5.

About LDAP Schema 2

LDAP Schema 2 is a set of provisioning definitions that describes the types of information that can be stored as entries by using the Directory Server LDAP.

The native mode uses search templates to search the Directory Server LDAP. Once the domain is found by using the domain search template, the user or group search templates are used to find a specific user or group.

You should use native mode if you are installing Contacts Server for the first time and you do not have other applications in your deployment that are dependent on a two-tree provisioning model.

Note: If you have an existing deployment that uses Schema 1, and you want to integrate other Communications Suite products, you should migrate your directory to Schema 2. Refer to ***Sun Java System Communications Services 6 2005Q4 Schema Migration Guide*** for information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2.

About LDAP Schema 1

LDAP Schema 1 is a provisioning schema that consists of both an Organization Tree and a DC Tree. In Schema 1, when a search is conducted for user or group entries, it looks at the user's or group's domain node in the DC Tree and extracts the value of the **inetDomainBaseDN** attribute. This attribute holds a DN reference to the organization subtree containing the actual user or group entry.

About LDAP Schema 2 Compatibility Mode

Schema 2 compatibility mode is an interim mode between Schema 1 and Schema 2 native mode. Schema 2 compatibility mode supports both schemas and enables you to retain the existing two-tree design you already have.

Use Schema 2 Compatibility if you have existing applications that require Schema 1, but you also need functionality that requires Schema 2.

Note: Schema 2 compatibility mode is provided as a convenience in migrating to the Schema 2 Native mode. Do not use Schema 2 compatibility mode as your final schema choice. The migration process from Schema 1 to Schema 2 compatibility mode and then finally to Schema 2 native mode is more complex that simply migrating from Schema 1 to Schema 2 native mode. See ***Sun Java System Communications Services 6 2005Q4 Schema Migration Guide*** for more information.

Attribute Indexes Created by the comm_dssetup.pl Script

Attribute indexes improve the performance of search algorithms. The **comm_dssetup.pl** script offers you the choice to index attributes.

Table B–2 lists all the attributes the **comm_dssetup.pl** script indexes, grouped by suffix category. It also lists the type of indexes created for each attribute. For more information about Directory Server indexing, see the Directory Server documentation at:

http://docs.oracle.com/cd/E20295_01/index.htm

Table B–2 Attributes Indexed by comm_dssetup.pl

Suffix	Attributes Indexed	Type of Indexes Added
User/Group	**mail**	**pres, eq, approx, sub**
User/Group	**mailAlternateAddress**	**pres, eq, approx, sub**
User/Group	**mailEquivalentAddress**	**pres, eq, approx, sub**
User/Group	**mailUserStatus**	**pres, eq**
User/Group	**member**	**eq**
User/Group	**ou**	**pres**
User/Group	**cosspecifier**	**pres**
User/Group	**groupid**	**pres, eq, sub**
User/Group	**icsCalendar**	**pres, eq, approx, sub**
User/Group	**icsCalendarOwned**	**pres, eq, approx, sub**
User/Group	**uniqueMember**	**eq**
User/Group	**memberOf**	**eq, sub**
User/Group	**cn**	**eq**
User/Group	**mgrpUniqueId**	**eq**
User/Group	**deleted**	**pres, eq**
User/Group	**davuniqueid**	**pres, eq**
User/Group	**inetCos**	**eq**
User/Group (additional for Schema 2)	**inetDomainBaseDN**	**pres, eq**
User/Group (additional for Schema 2)	**sunPreferredDomain**	**pres, eq**
User/Group (additional for Schema 2)	**associatedDomain**	**pres, eq**
User/Group (additional for Schema 2)	**o**	**pres, eq**
User/Group (additional for Schema 2)	**mailDomainStatus**	**pres, eq**
User/Group (additional for Schema 2)	**sunOrganizationAlias**	**pres, eq**
DC Tree (for Schema 1)	**inetDomainBaseDN**	**pres, eq**
DC Tree (for Schema 1)	**mailDomainStatus**	**pres, eq**
DC Tree (for Schema 1)	**inetCanonicalDomainName**	**pres, eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfManagedGroup**	**pres, eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfPAB**	**pres, eq**
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**memberOfPABGroup**	**pres,eq**

Table B–2 (Cont.) Attributes Indexed by comm_dssetup.pl

Suffix	Attributes Indexed	Type of Indexes Added
Personal Address Book (PAB) (**o=pab**) Note: For old Address Book	**un**	**eq**
New PAB (**o=PiServerDb**)	**displayname**	**pres**, **eq**, **sub**
New PAB (**o=PiServerDb**)	**MemberOfPiBook**	**eq**
New PAB (o=PiServerDb)	**MemberofPiGroup**	**eq**
o=mlusers for future mailserv feature	**mail**	**eq**
o=mlusers for future mailserv feature	**mlsubListIdentifier**	**eq**
o=mlusers for future mailserv feature	**mlsubMail**	**eq**

To add additional indexes on your own, see the Directory Server documentation.

Running the comm_dssetup.pl Script

You can run the **comm_dssetup.pl** script in either interactive or silent mode. Interactive mode is described in "Running the comm_dssetup.pl Script in Interactive Mode".

Running the comm_dssetup.pl Script in Silent Mode

To run the **comm_dssetup.pl** script in silent mode:

1. On the host where Directory Server is installed, log in as or become the superuser (**root**).
2. Start Directory Server, if necessary.
3. Change to the directory where you installed or copied the Directory Server Setup **comm_dssetup.pl** script.
4. Run the script followed by the silent mode options.

   All options are required. For more information, see "Silent Mode Options".

   ```
   /usr/bin/perl comm_dssetup.pl
   [ -i yes | no ] [ -R yes | no ] [ -c DirectoryServerRoot ]
   [ -d DirectoryInstance ] [ -r DCTreeSuffix ]
   [ -u UserGroupSuffix ] [ -s yes | no ] [ -D DirectoryManagerDN ]
   [ -j DirectoryManagerPasswordFile ] [ -b yes | no ]
   [ -t 1 | 1.5 | 2 ] [ -m yes | no ] [-S PathtoSchemaFiles ]
   ```

   The script creates the following LDIF file and shell script to update the LDAP indexes and schema:

   - **/var/tmp/*dssetup_timestamp*/dssetup.ldif**
   - **/var/tmp/*dssetup_timestamp*/dssetup.sh**
5. If you answered **no** to the **-R** and **-m** options, you must manually run the **comm_dssetup.sh** script that was created.

If you answered **yes** to the **-R** and **-m** options, the **dssetup.sh script** is run automatically.

Silent Mode Options

Table B–3 table describes the **comm_dssetup.pl** silent mode options.

Table B–3 comm_dssetup.pl Silent Mode Options

Option and Argument	Description
-i yes | **no**	Specifies whether to configure new indexes. **yes** - Add new Directory Server indexes. **no** - Do not add indexes.
-R yes | **no**	Specifies whether to reindex automatically. **yes** - Reindex without prompting the user. **no** - Do not reindex without prompting the user. The **-m** option must also be specified for **yes** for the **-R** option to take effect.
-c *DirectoryServerRoot*	Specifies the Directory Server root path, for example, **/var/opt/sun/directory**.
-d *DirectoryInstance*	Specifies the Directory Server instance subdirectory under the Directory Server root path, for example, **slapd-varrius**.
-r *DCTreeSuffix*	Specifies the DC tree root suffix (for Schema 1 and Schema 2 compatibility modes only), for example, **o=internet**.
-u *UserGroupSuffix*	Specifies the user and group root suffix, for example, **o=usergroup**.
-s yes | no	Specifies whether to update the schema. **yes** - Update the schema. **no** - Do not update schema.
-D *DirectoryManagerDN*	Specifies the Directory Manager Distinguished Name (DN), for example, **"cn=Directory Manager"**. The value must be enclosed by double quotation marks (" ") to enable the **comm_dssetup.pl** script to interpret a value with a space correctly.
-j *DirectoryManagerPasswordFile*	Specifies the file containing the Directory Manager DN password.
-b yes | **no**	Specifies to use this Directory Server for users and groups. **yes** - Use this directory to store both configuration and user group data. **no** - Use this directory to store only configuration data.
-t 1 | **1.5** | **2**	Specifies the schema version.
-m yes | **no**	Specifies whether to modify the Directory Server. **yes** - Modify the Directory Server without prompting the user. **no** - Do not modify the Directory Server without prompting the user.
-S *PathtoSchemaFiles*	Specifies the path to the directory where the schema files are located for example, **./schema**.

C

Contacts Server Configuration Scripts

This appendix provides information about the Oracle Communications Contacts Server configuration scripts.

init-config Script

The **init-config** script enables you to perform an initial configuration of your Contacts Server deployment.

Table C–1 shows the **init-config** options.

Table C–1 init-config Options

Option	Description
-d	Writes an LDIF file instead of making changes to the Directory Server data.
-f *statefile*	Uses the *statefile* for setting input values. Use the **nab.defaults** file as an example.
-i *hostname*	Uses *hostname* as the FQDN of current host.
-l	Uses **/bin/hostname** for name of host. **/bin/hostname** must return an FQDN.
-s	Performs a silent install, requires the `-f` option.
-S *statefile*	Saves state of configurator input to a named *statefile*.
-v	Enables verbose mode.
-W	Use this option for WebLogic Server. It is applicable for the initial configuration. **Note**: If you do not use this option, GlassFish Server is used as the default application server.

extractSSLArgs.sh Script

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

Options

The following table lists the options for validating and storing Oracle WebLogic Server SSL details.

Table C–2 WebLogic Server Options

Option	Description
-u	WebLogic Server Administrator User ID
-p	WebLogic Server Administrator User Password
-l	WebLogic Server Administrator URL *t3|t3s://FQDN:Weblogic AdminServer (SSL|NonSSL)Port* For example, **t3://hostname.com:7001** Or **t3s://hostname.com:7002**
-r	Runtime directory under which **/config/davadmin.properties** file resides. For example, **/var/opt/sun/comms/nabserver/config/davadmin.properties**. You can use this option only on the existing Contacts Server deployment which is already running on WebLogic Server. Use this option if WebLogic Server's keystore settings are modified after Contacts Server is deployed and those settings have to be updated on Contacts Server Configuration.

Running the Script for a Fresh Installation

- Validate the SSL and Keystore configurations on WebLogic Server setup
- Perform the steps in the "Installing and Configuring WebLogic Server" section to set up WebLogic Server in a secure mode
- Keep the configuration details of WebLogic Server handy

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

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

   ```
   cd Weblogic_Domain/bin

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

2. Set the **WLST_PROPERTIES** environment variable depending on the selected Oracle WebLogic Server keystore configuration.

   - If the **CustomIdentityandCustomTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_PROPERTIES="-Dweblogic.security.TrustKeyStore=CustomTrust ,
     -Dweblogic.security.CustomTrustKeyStoreFileName=Weblogic_
     Domain/mytrust.jks"
     ```

     where *Weblogic_Domain/mytrust.jks* is the location of truststore file location.

   - If the **CustomIdentityandJavaStandardTrust** keystore option is configured as the Oracle WebLogic Server keystore configuration, set the **WLST_PROPERTIES** variable to:

     ```
     export WLST_
     PROPERTIES="-Dweblogic.security.TrustKeyStore=JavaStandardTrust"
     ```

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

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

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

Running the Script to Repair Keystore Information

If you modify keystore settings of the in-production WebLogic Server on which Contacts Server is deployed, you must update the previously supplied SSL Keystore details on Contacts Server. By using the **extractSSLArgs.sh -r** option, you can update the keystore information.

1. Ensure that WebLogic Server Keystore and SSL configurations are modified and you have the latest configuration details.
2. Open a new terminal and prepare the terminal by sourcing the **setDomainEnv.sh** script of the Oracle WebLogic Server domain:

   ```
   cd Weblogic_Domain/bin
   source ./setDomainEnv.sh   OR  . ./setDomainEnv.sh
   ```

3. Set the **WLST_PROPERTIES** environment variable. See step 2 in "Running the Script for a Fresh Installation". Ensure to enter the latest keystore details.
4. Run the following script:

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

   For example,

   sh extractSSLArgs.sh -u weblogic -p weblogic123 -l t3s://myhost.domain.com:7002 -r /var/opt/sun/comms/nabserver

5. Restart WebLogic Server.

   Weblogic_Domain/**config/.wls_sslargs** and *nabserver_runtime_dir*/**config/davadmin.properties** files are updated according to the Keystore modifications performed on WebLogic Server.

Note: *ContactsServer_home*/**config/** contains the **davadmin.properties** file. CLI **davadmin** utility of the product uses the **davadmin.properties** file. Therefore, any modifications performed on WebLogic Server for keystore settings must reflect in the **davadmin.properties** file.

Database Installation Scripts

Contacts 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 Contacts 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 C–3 shows the **config-mysql** ***options***:

Table C–3 config-mysql Options

Option	Description
--help | **-?**	Prints help.
--contacts | **-c**	Configures the contacts 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:

```
config-mysql -s -u -c
```

To set up just a new contacts 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 Contacts Server database.

To run the **config-mysql** script:

1. Copy the **config-mysql** and **Util.pm** scripts from a host where Contacts Server is installed to the host where MySQL is installed.

   Both scripts are located in the ***ContactsServer_home*/tools/unsupported/bin** directory.

2. On the host where MySQL is installed, as **root**, launch the **config-mysql** script with options **-s**, **-u**, and **-c**.

   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), and contacts 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 "Change the root password?" question.

     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 contacts 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 Contacts 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 Contacts Server. See "Preparing Oracle Database for Use with Contacts Server" 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 C–4 shows the **config-oracle** *options*:

Table C–4 config-oracle Options

Option	Description
--help | **-?**	Prints help.

Table C–4 (Cont.) config-oracle Options

Option	Description
--contacts | **-c**	Configures the contacts database.

To set up the default 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 host where Contacts Server is installed to the host where Oracle Database is installed.

   Both scripts are located in the ***ContactsServer_home*/tools/unsupported/bin** directory.

2. On the host where Oracle Database is installed, as **root**, enter the following command:

   ```
   config-oracle -c
   ```

3. Enter the Oracle SYS user password.
4. Enter the contacts database user name and password.

   The script outputs the list of tasks to be performed.

5. Enter **Y** to proceed.

   The script creates the contacts database user and schemas.