Oracle® Communications Messaging Server

Installation and Configuration Guide

Release 8.0.2

E72264-01

October 2017

ORACLE®

Oracle Communications Messaging Server Installation and Configuration Guide, Release 8.0.2

E72264-01

Contents

Preface

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

Audience

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

- Messaging protocols, such as IMAP and SMTP
- Oracle Directory Server Enterprise Edition and LDAP
- System administration and networking

Related Documents

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

- *Messaging Server System Administrator's Guide*: Provides instructions for administering Messaging Server.
- *Messaging Server Reference*: Provides additional information for using and configuring Messaging Server.
- *Messaging Server Release Notes*: Describes the fixes, known issues, troubleshooting tips, and required third-party products and licensing.
- *Messaging Server Security Guide*: Provides guidelines and recommendations for setting up Messaging Server in a secure configuration.
- *Messaging Server Installation and Configuration Guide for Cassandra Message Store*: Provides instructions for installing and configuring the Cassandra message store.
- *Messaging Server MTA Developer's Reference*: Describes the Messaging Server MTA SDK.

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 Messaging Server Installation and Configuration Overview

This chapter provides an overview of the Oracle Communications Messaging Server installation and configuration process.

> **Note:** Starting with the release of Messaging Server 8.0.2, you can choose to use either classic message store or "Cassandra" message store, which is based on DataStax Cassandra Database. You cannot use both message stores in the same deployment. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information about installing Cassandra message store.

Overview of Messaging Server Installed Components

During the installation process, you install and configure Messaging Server.

Messaging 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`

Overview of the Messaging 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 Messaging Server is dependent, such as Directory Server.
4. Prepare the Directory Server schema by installing and running the most current **comm_dssetup** script from the Messaging Server distribution.
5. Install and configure Messaging Server.
6. Perform post-installation configuration tasks.
7. Verify the installation.

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

Messaging Server Installation Options

You install Messaging Server by running the installer 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.

For more information, see the discussion on running a non-interactive session in "Installing Messaging Server in Silent Mode".

Ensuring a Successful Messaging Server Installation

Only qualified personnel should install the product. You must be familiar with the UNIX operating system. 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, 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 Messaging Server Directory Placeholders

Placeholder	**Directory**
MessagingServer_home	Specifies the installation location for the Messaging Server software. The default is **/opt/sun/comms/messaging64**.
InstallRoot	Specifies the installation location for other Communications Suite software. The default is **/opt/sun/comms**.
ConfigRoot	Specifies the location of the configuration files. The default is **/var/opt/sun/comms/messaging64/config**.
DataRoot	Specifies the location of the data files. The default is **/var/opt/sun/comms/messaging64**.

2

Planning Your Messaging Server Installation

This chapter provides information about planning your Oracle Communications Messaging Server installation. It also describes the Messaging Server logical architecture.

For information about planning a Cassandra message store installation, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

About Messaging Server

Messaging Server is a high-performance, highly secure messaging platform that can scale from thousands to millions of users. It provides extensive security features that help ensure the integrity of communications through user authentication, session encryption, and the appropriate content filtering to reduce spam and viruses. With Messaging Server, enterprises and service providers can provide secure, reliable messaging services for entire communities of employees, partners, and customers.

Messaging Server provides a powerful and flexible solution to the email needs of enterprises and messaging hosts of all sizes by using open Internet standards.

Determining Your Messaging Server Network Infrastructure Needs

Your network infrastructure is the underlying foundation of the system. It forms the services that create the operating makeup of your network. In a Messaging Server deployment, determining your network infrastructure from the project ensures that you will have an architecture that can scale and grow.

The topics in this section include:

- Understanding Your Existing Network
- Understanding Network Infrastructure Components
- Planning Your Network Infrastructure Layout

Understanding Your Existing Network

You need to understand your existing network infrastructure to determine how well it can meet the needs of your deployment goals. By examining your existing infrastructure, you identify if you need to upgrade existing network components or purchase new network components. You should build up a complete map of the existing network by covering these areas:

- Physical communication links, such as cable length, grade, and so forth

- Communication links, such as analog, ISDN, VPN, T3, and so forth, and available bandwidth and latency between sites
- Server information, including:
  - Host names
  - IP addresses
  - Domain Name System (DNS) server for domain membership
- Locations of devices on your network, including:
  - Hubs
  - Switches
  - Modems
  - Routers and bridges
  - Proxy servers
- Number of users at each site, including mobile users

After completing this inventory, you need to review that information in conjunction with your project goals to determine what changes are required so that you can successfully deliver the deployment.

Understanding Network Infrastructure Components

The following common network infrastructure components have a direct impact upon the success of your deployment:

- Routers and Switches
- Firewall Access Control
- Load Balancers
- Cassandra Storage
- Classic Storage
- Domain Name System (DNS)

Routers and Switches

Routers connect networks of your infrastructure, enabling systems to communicate. You need to ensure that the routers have spare capacity after the deployment to cope with projected growth and usage.

In a similar vein, switches connect systems within a network.

Routers or switches running at capacity tend to induce escalating bottlenecks, which result in significantly longer times for clients to submit messages to servers on different networks. In such cases, the lack of foresight or expenditure to upgrade the router or switch could have a personnel productivity impact far greater than the cost.

Firewall Access Control

Firewalls sit between a router and application servers to provide access control. Firewalls were originally used to protect a trusted network (yours) from the untrusted network (the Internet). These days, it is becoming more common to protect application servers on their own (trusted, isolated) network from the untrusted networks (your network and the Internet).

Router configurations add to the collective firewall capability by screening the data presented to the firewall. Router configurations can potentially block undesired services (such as NFS, NIS, and so forth) and use packet-level filtering to block traffic from untrusted hosts or networks.

In addition, when installing a Sun server in an environment that is exposed to the Internet, or any untrusted network, reduce the Oracle Solaris software installation to the minimum number of packages necessary to support the applications to be hosted. Achieving minimization in services, libraries, and applications helps increase security by reducing the number of subsystems that must be maintained. The Oracle Solaris Security Toolkit provides a flexible and extensible mechanism to minimize, harden, and secure Oracle Solaris systems.

Your Site Security Policy should provide direction on such issues.

Load Balancers

Use load balancers to distribute overall load on your Web or application servers, or to distribute demand according to the kind of task to be performed. If, for example, you have a variety of dedicated applications and hence different application servers, you might use load balancers according to the kind of application the user requests.

If you have multiple data centers, you should consider geographic load balancing. Geographic load balancing distributes load according to demand, site capacity, and closest location to the user. If one center should go down, the geographic load balancer provides failover ability.

For load balancers on Web farms, place the hardware load balancers in front of the servers and behind routers because they direct routed traffic to appropriate servers. Software load balancing solutions reside on the Web servers themselves. With software solutions, one of the servers typically acts a traffic scheduler.

A load balancing solution is able to read headers and contents of incoming packets. This enables you to balance load by the kind of information within the packet, including the user and the type of request. A load balancing solution that reads packet headers enables you to identify privileged users and to direct requests to servers handling specific tasks.

You need to investigate how dynamically the load balancer communicates with all the servers it caters to. Does the scheduler ping each server or create "live" agents that reside on the servers to ascertain load data? You should also examine how the load balancer parses TCP packets. Pay attention to how quickly the load balancer can process a packet. Some load balancers will be more efficient than others. Load balancer efficiency is typically measured in throughput.

Cassandra Storage

For Cassandra message store deployments, it is recommended to use solid-state drives (SSDs) for nodes running DataStax Cassandra database.

For more information, see the DataStax documentation about selecting hardware at:

http://docs.datastax.com/en/dse-planning/doc/planning/planningHardware.html

Classic Storage

Deploying SANs for classic message store can represent a decrease in the time to recover from a non-functional server as the machine can be replaced without having to relocate the storage drives.

Use these questions to evaluate if your deployment storage requirements would be best served through a SAN:

- Are reads or writes more prevalent?
- Do you need high I/O rate storage? Is striping the best option?
- Do you need high uptime? Is mirroring the best option?
- How is the data to be backed up? When is it going to be backed up?

For more information, see the discussion on planning your storage in *Messaging Server System Administrator's Guide*.

Domain Name System (DNS)

Servers which make heavy usage of DNS queries should be equipped with a local caching DNS server to reduce lookup latency as well as network traffic.

When determining your requirements, consider allocating host names for functions such as mailstore, mail-relay-in, mail-relay-out, and so forth. You should consider this policy even if the host names all are currently hosted on one machine. With services configured in such a way, relocation of the services to alternate hardware significantly reduces the impacts of the change.

Planning Your Network Infrastructure Layout

In deriving your infrastructure topology, you must consider the following topics:

- Demilitarized Zone (DMZ)
- Intranet
- Internal Network
- Firewall Configuration
- Mobile Users

Demilitarized Zone (DMZ)

These days, most company networks are configured for a DMZ. The DMZ separates the corporate network from the Internet. The DMZ is a tightly secured area into which you place servers providing Internet services and facilities (for example, web servers). These machines are hardened to withstand the attacks they might face.

Progressively, DMZ implementations have moved the segment behind the firewall as firewall security and facilities have increased in robustness. However, the DMZ still remains segmented from the internal networks. You should continue to locate all machines hosting Web servers, FTP servers, mail servers, and external DNS on a DMZ segment. Note that MTAs, Webmail (mshttpd) servers, and MMPs belong in the DMZ. The IMAP servers and Cassandra servers generally should not be in the DMZ, but rather in a separate (more restricted) network that can only be accessed by the MTA/mshttpd/MMP machines in the DMZ.

A simpler network design might only define separate DMZ segments for Internet services, VPN access, and remote access. However, security issues exist with VPN and remote access traffic. You need to separate appropriate connections of these types from the rest of the network.

The firewall providing the DMZ segmentation should allow only inbound packets destined to the corresponding service ports and hosts offering the services within the DMZ. Also, limit outbound initiated traffic to the Internet to those machines requiring access to the Internet to carry out the service they are providing (for example, DNS and mail). You might want to segment an inbound-only DMZ and an outbound-only DMZ, with respect to the type of connection requests. However, given the potential of a denial-of-service attack interrupting DNS or email, consider creating separate inbound and outbound servers to provide these services. Should an email-based Trojan horse or worm get out of control and overrun your outbound mail server, inbound email can still be received. Apply the same approach to DNS servers.

Intranet

The DMZ provides a network segment for hosts that offer services to the Internet. This design protects your internal hosts, as they do not reside on the same segment as hosts that could be compromised by an external attack. Internally, you also have similar services to offer (Web, mail, file serving, internal DNS, and so on) that are meant solely for internal users. Just as the Internet services are segmented, so too, are the internal services. Separation of services in this manner also permits tighter controls to be placed on the router filtering.

Just as you separate the Internet-facing services into the DMZ for security, your private internal services should reside in their own internal DMZ. In addition, just as multiple DMZs can be beneficial-depending on your services and your network's size-multiple intranets might also be helpful.

The firewall rules providing the segmentation should be configured similarly to the rules used for the DMZ's firewall. Inbound traffic should come solely from machines relaying information from the DMZ (such as inbound email being passed to internal mail servers) and machines residing on the internal network.

Internal Network

The segments that remain make up your internal network segments. These segments house users' machines or departmental workstations. These machines request information from hosts residing on the intranet. Development, lab, and test network segments are also included in this list. Use a firewall between each internal network segment to filter traffic to provide additional security between departments. Identify the type of internal network traffic and services used on each of these segments to determine if an internal firewall would be beneficial.

Machines on internal networks should not communicate directly with machines on the Internet. Preferably, these machines avoid direct communication with machines in the DMZ. Ultimately, the services they require should reside on hosts in the intranet. A host on the intranet can in turn communicate with a host in the DMZ to complete a service (such as outbound email or DNS). This indirect communication is acceptable.

Firewall Configuration

In addition to the typical packet-filtering features, most firewalls provide features to prevent IP spoofing. Use IP-spoofing protection whenever possible.

For instance, if there is only one entry point into your network from the Internet and a packet is received from the Internet with a source address of one of your internal machines, it was likely spoofed. Based on your network’s topology, the only packets containing a source IP address from your internal machines should come from within the network itself, not from the Internet. By preventing IP spoofing, this possibility is eliminated, and the potential for bypassing IP address-based authorization and the other firewall-filtering rules is reduced. Use the same IP-spoofing protection on any internal firewall as well.

Mobile Users

When you have remote or mobile users, pay attention to how you will provide them access to the facilities. Will there be any facilities they cannot access? What kind of security policies do you need to address? Will you require SSL for authentication? Also, examine whether your mobile user population is stable or is expected to increase over time.

Messaging Server Front-End and Back-End Components

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

- **Message Transfer Agent (MTA).** The MTA front-end component receives, routes, transfers, and delivers mail messages using the SMTP protocol. The MTA delivers messages to a local mailbox or to another MTA.
- **Webmail Server.** The Webmail Server front-end component provides email services to Convergence clients by using HTTP protocol.
- **Messaging Multiplexor (MMP).** The MMP front-end component enables horizontal scaling of Message Store notifications by routing connections to the machines with affinity for a given user.
- **Message Store.** The Message Store back-end component stores, retrieves, and manipulates messages for mail clients.
- **LDAP Directory.** The LDAP Directory back-end component stores and retrieves information about the user base. It stores user and group aliases, mailhost information, delivery preferences, and so on.

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

Planning Your Messaging Server Installation

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

- Planning for Multiple Messaging Server Hosts
- Planning for Virus Scanning
- Planning a Messaging Server Topology
- Planning a Messaging Server Sizing Strategy

Planning for Multiple Messaging Server Hosts

Using multiple Messaging Server hosts can help you:

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

Planning for Virus Scanning

Messaging Server requires use of third-party anti-spam and anti-virus technology for real world deployments. The recommended integration method is to call this technology from the Messaging Server MTA by using the milter protocol. For more information about the milter implementation, see *Messaging Server Reference*. If you choose to configure virus scanning, decide whether to use an existing Messaging Server MTA, or to deploy a dedicated MTA-only Messaging Server installation to scan for viruses. For more information, see the discussion on configuring virus scanning in *Messaging Server System Administrator's Guide*.

Planning a Messaging Server Topology

Messaging Server allows for different types of messaging topology: central topology, distributed topology, hybrid topology, and system provider topology. Before choosing a messaging topology, you must determine the messaging services you need to provide at each location within your organization. For more information, see the discussion on which messaging topology will suit your organization in "Designing a Messaging Server Topology".

Planning a Messaging Server Sizing Strategy

Before installing Messaging Server, you must consider a sizing process in order to provide optimal performance, scalability, and reliability. You must determine the data you need to size for your Messaging Server deployment. For more information, see the discussion on determining the correct sizing strategy for your organization in "Planning a Messaging Server Sizing Strategy".

System Deployment Planning

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

- Planning for High Availability
- Using Load Balancing
- Planning Backup Strategies

Planning for High Availability

Note: For Cassandra message store, the Cassandra database is designed to be highly available with no single point of failure. Thus, you do not need to install additional cluster software to create a high availability solution for the message store itself.

You can configure Messaging Server components to be highly available. Messaging Server supports three different high availability solutions, Oracle Solaris Cluster, Veritas Cluster Server (VCS), and Oracle Clusterware. Refer to the documentation for those products for installation and configuration instructions. For more information, see the discussion on high availability in "Configuring Messaging Server for High Availability".

Using Load Balancing

Load balancers balance network connections uniformly or by algorithm across multiple servers. You cannot use load balancers on the Message Store or directory masters. You can use them for connections to MMPs, Convergence, MTAs, and directory consumers.

For more information, see the discussion on load balancers in "Developing a Messaging Server Architecture".

Planning Backup Strategies

Note: The backup strategies described in this section pertain to classic message store. The **imsbackup** utility can be used to archive a user's data when that user's account is decommissioned. Otherwise, Cassandra's built-in replication and disaster recovery capabilities are the backup strategy for Cassandra message store.

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

The three ways to back up Messaging Server data are:

- **imsbackup** utility
- Solstice backup (Legato Networker)
- Oracle Solaris ZFS snapshots

For more information, see the discussion on backup strategies in *Messaging Server System Administrator's Guide*.

Messaging Server Logical Architecture

Note: For a discussion of Cassandra message store logical architecture, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

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

- **Single-tiered Messaging Server architecture:** You can deploy all components on a single host. There is no separation between access and data layers. The MTA, Message Store, and sometimes the Directory Server are installed in one layer.
- **Two-tiered Messaging Server architecture:** You can deploy Messaging Server with the front-end components installed on a separate host and the database back end installed on another host. A two-tiered architecture splits the Messaging Server deployment into two layers: an access layer and data layer.
- **Two-tiered, multiple server Messaging 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.

For more information, see the discussion on designing a Messaging Server architecture in "Developing a Messaging Server Architecture".

About Installing a Secure System

You can secure your Messaging Server infrastructure by first determining your firewall or DMZ architecture. You should at least configure Secure Sockets Layer (SSL) on Messaging Server front-end hosts. You can also configure SSL on database back-end hosts. You must also protect Messaging Server individual components.

For example, since Webmail Server supports unencrypted and encrypted (SSL) communication with mail clients, you might want to use a firewall between the Message Store and mail clients for added security.

For information, see the discussion on secure installation and configuration of Messaging Server in *Messaging Server Security Guide*.

3

Developing a Messaging Server Architecture

This chapter provides information on how to design the architecture of your Oracle Communications Messaging Server, as well as information on how Messaging Server components are distributed across hardware and software resources.

For information about designing a Cassandra message store architecture, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

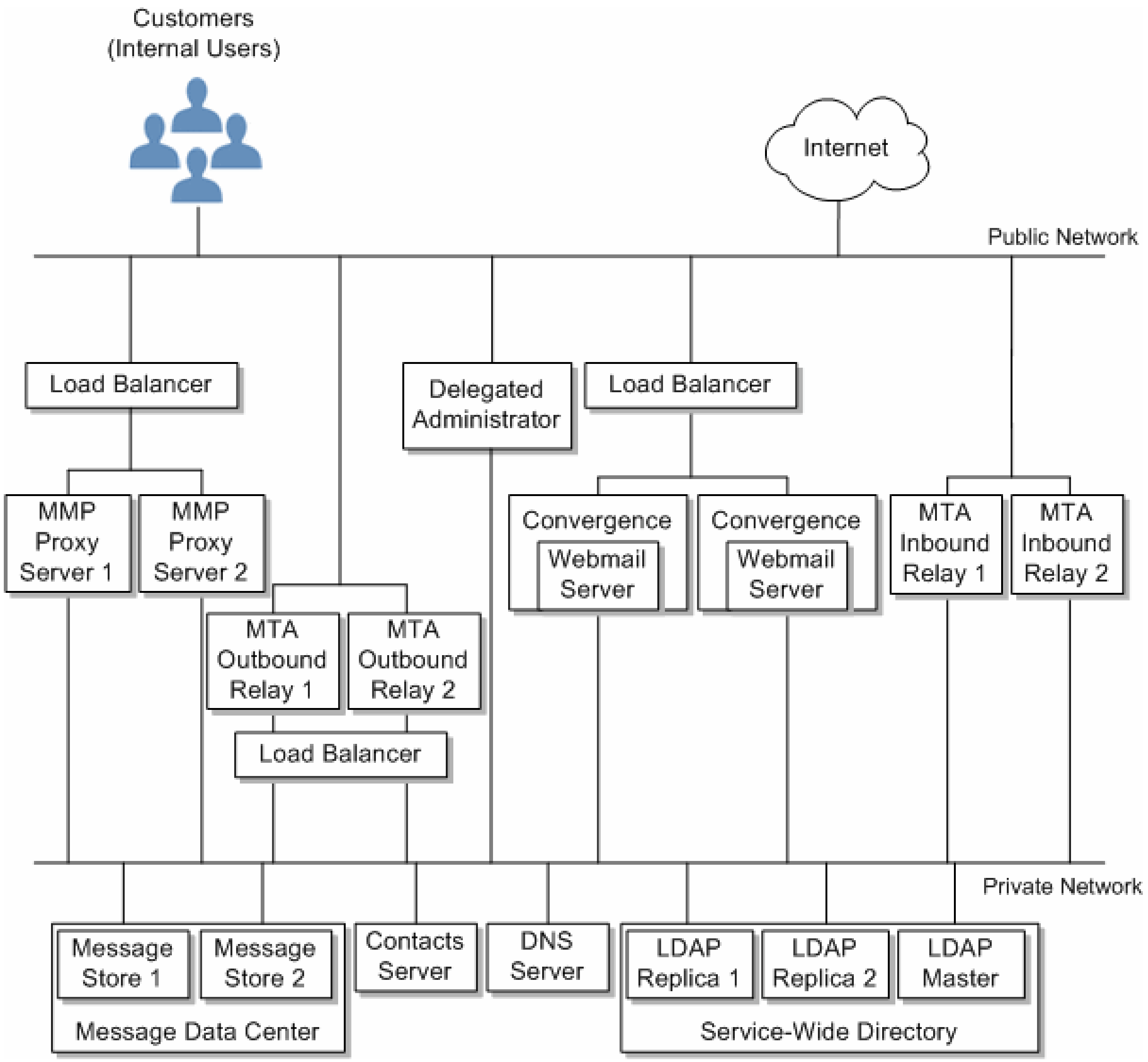
Understanding the Two-tiered Messaging Architecture

A two-tiered messaging architecture provides the optimum design for scalability and reliability. Instead of having a single host run all the components of a messaging system, a two-tiered architecture separates the components onto different machines. These separate components perform specific specialized functions. As the load for a particular functional component increases-for example, more Message Storage is required, or more outbound relaying is needed-you can add more servers to handle the larger loads.

The two-tiered architecture consists of an access layer and a data layer. The access layer is the portion of the architecture that handles delivery, message access, user login, and authentication. The data layer is the portion of the architecture that holds all the data. This includes the LDAP master servers and Messaging Server machines that are configured to store user messages.

Figure 3–1 shows an example two-tiered architecture.

Figure 3–1 Two-Tiered Messaging Server Architecture



The following describes each of these functional pieces.

Public Access Network. The network connecting the Messaging Server to internal users and the Internet. Each deployment defines its own network requirements; however, the basic Messaging Server requirement is connectibility to end users and the Internet using standard protocols such as SMTP, POP, IMAP, and HTTP.

Private Data Network. This network provides secure connectivity between the public access network and Messaging Server data. It consists of a secure access layer and a data layer, which includes the service-wide directory, the message data center, and the contact server.

LDAP directory server. Directory server used for storing and retrieving information about the user base. It stores user and group aliases, mailhost information, delivery preferences, and so on. Depending on your design requirements, there could be more than one identical directory for the system. Figure 3–1 shows a master directory and two replicas. An LDAP directory server is provided as part of the Messaging Server product. If desired, you can use data from an existing Directory Server. The data format of the existing directory must also be compliant with the Messaging Server schema.

Message Store. Holds and stores user mail. Sometimes referred to as a "back end." The Message Store also refers to the Message Access Components such as the IMAP server and the POP server. Figure 3–1 shows a deployment that has two message stores. You can add more stores as needed.

> **Note:** When deploying Cassandra message store, the store architecture changes. Instead of a "Message Data Center" with "Message Store Host 1" and "Message Store Host 2," you deploy store affinity groups with message access hosts running IMAP and LMTP processes. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Contacts Server. Stores and retrieves users' addresses in an LDAP server, which can be the same server or a different server from the LDAP server described above.

DNS server. Maps host names to IP addresses. The DNS server determines what host to contact when routing messages to external domains. Internally, DNS maps actual services to names of machines. The DNS server is not part of the Messaging Server product. You must install an operating DNS server prior to installing Messaging Server.

Load Balancer. Balances network connections uniformly or by algorithm across multiple servers. Using load balancers, a single network address can represent a large number of servers, eliminating traffic bottlenecks, allowing management of traffic flows and guaranteeing high service levels. Figure 3–1 shows load balancers for the MMPs and the MTA submission servers. Load balancers are not part of the Messaging Server. You cannot use load balancers on the Message Store or directory masters. You use them for connections to MMPs, Convergence, MTAs submission servers (port465/587), and directory consumers.

MTA Inbound Relay. MTA dedicated to accepting messages from external (Internet) sites and routing those messages to internal hosts and the local Message Store server. Because this is the first point of contact from the outside, the MTA inbound relay has the added responsibility of guarding against unauthorized relaying, spam filtering, and denial of service attack. You can use MX records to balance incoming mail traffic. See "Mail Exchange (MX) Records" for more information. MTA relays on configured to use port 25 get automatic load balancing from DNS MX records. A load-balancer is fine for a Message Submission MTA (port 465/587).

MTA Outbound Relay. MTA that only receives mail from internal or authenticated users and routes those messages to other internal users or to external (Internet) domains. While a single machine can be an inbound relay as well as an outbound relay, in a large scale Internet-facing deployment, separate these functions to two separate machines. This way, internal clients sending mail do not have to compete with inbound mail from external sites.

Delegated Administrator Server. Provides a GUI management console for administrators, enabling more advanced administrative tasks, such as adding and deleting users.

Messaging Multiplexor or **MMP.** Enables scaling of the Message Store across multiple physical machines by decoupling the specific machine that contains a user's mailbox from its associated DNS name. Client software does not have to know the physical machine that contains its Message Store. Thus, users do not need to change the name of their host message store every time their mailbox is moved to a new machine. When POP or IMAP clients request mailbox access, the MMP forwards the request to the Messaging Server system containing the requested mailbox by looking in the directory service for the location of the user's mailbox. When you use multiple MMPs, they should be located behind a load balancer.

Webmail Server or mshttpd daemon. Provides email services to Convergence clients by using HTTP. In previous versions of Messaging Server, the Webmail Server accessed the Message Store directly. Now, the Webmail Server accesses the Message Store through the IMAP server. Such an architecture enables Convergence clients to access shared folders that are located in different back-end Message Stores. Additionally, there is no longer a requirement to install the Webmail Server on each back-end server. The Webmail Server can act as a front-end server performing multiplexing capabilities.

We only support mshttpd talking to our IMAP and our MTA. We only support Convergence talking to mshttpd. The WMAP protocol mshttpd exposes is not intended for use by third parties. The mshttpd ports (8990, 8991) should be fire-walled so only the Convergence front-ends can talk to it. Or the mshttpd service should be disabled as it is by default.

Two-tiered Architecture: Messaging Data Flow

This section describes the message flow through the messaging system. How the message flow works depends upon the actual protocol and message path.

Sending Mail: Internal User to Another Internal User

Synopsis: Internal User > Load Balancer > MTA Outbound Relay 1 or 2 > MTA Inbound Relay 1 or 2 > Message Store 1 or 2

Note: An increasingly more common scenario is to use LMTP to deliver mail directly from the outbound relay to the store. In a two-tiered deployment, you can make this choice.

Messages addressed from one internal user to another internal user (that is, users on the same email system) first go to a load balancer. The load balancer shields the email user from the underlying site architecture and helps provide a highly available email service. The load balancer sends the connection to either MTA Outbound Relay 1 or 2. The outbound relay reads the address and determines that the message is addressed to an internal user. The outbound relay sends the message to MTA Inbound Relay 1 or 2 (or directly to the appropriate message store if so configured). The MTA Inbound Relay delivers the message to the appropriate Message Store. The Message Store receives the message and delivers it to the mailbox.

Retrieving Mail: Internal User

Synopsis: Internal User > Load Balancer > MMP/Convergence Server 1 or 2 > Message Store 1 or 2

Mail is retrieved by using either POP or IMAP. The user connection is received by the load balancer and forwarded to one of the MMP or Convergence servers. The user then sends the login request to the access machine it is connected to. The access layer machine validates the login request and password, then sends the request over the same protocol designated by the user connection to the appropriate Message Store (1 or 2). The access layer machine then proxies for the rest of the connection between the client and servers.

Sending Mail: Internal User to an External (Internet) User

Synopsis: Internal User > Load Balancer > MTA Outbound Relay 1 or 2 > Internet

Messages addressed from an internal user to an external user (that is, users not on the same email system) go to a load balancer. The load balancer shields the email user from the underlying site architecture and helps provide a highly available email service. The load balancer sends the message to either MTA Outbound Relay 1 or 2. The outbound relay reads the address and determines that the message is addressed to an external user. The outbound relay sends the message to an MTA on the Internet.

Sending Mail: External (Internet) User to an Internal User

Synopsis: External User > MTA Inbound Relay 1 or 2 > Message Store 1 or 2

Messages addressed from an external user (from the Internet) to an internal user go to either MTA Inbound Relay 1 or 2 (a load balancer is not required). The inbound relay reads the address and determines that the message is addressed to an internal user. The inbound relay determines by using an LDAP lookup whether to send it to Message Store 1 or 2, and delivers accordingly. The appropriate Message Store receives the message and delivers it to the appropriate mailbox.

Understanding Horizontal and Vertical Scalability in Messaging Server

Scalability is the capacity of your deployment to accommodate growth in the use of messaging services. Scalability determines how well your system can absorb rapid growth in user population. Scalability also determines how well your system can adapt to significant changes in user behavior, for example, when a large percentage of your users want to enable SSL within a month.

This section helps you identify the features you can add to your architecture to accommodate growth on individual servers and across servers. The following topics are covered:

- Planning for Horizontal Scalability
- Planning for Vertical Scalability

Planning for Horizontal Scalability

Horizontal scalability refers to the ease with which you can add more servers to your architecture. As your user population expands or as user behavior changes, you eventually overload resources of your existing deployment. Careful planning helps you to determine how to appropriately scale your deployment.

If you scale your deployment horizontally, you distribute resources across several servers. There are two methods used for horizontal scalability:

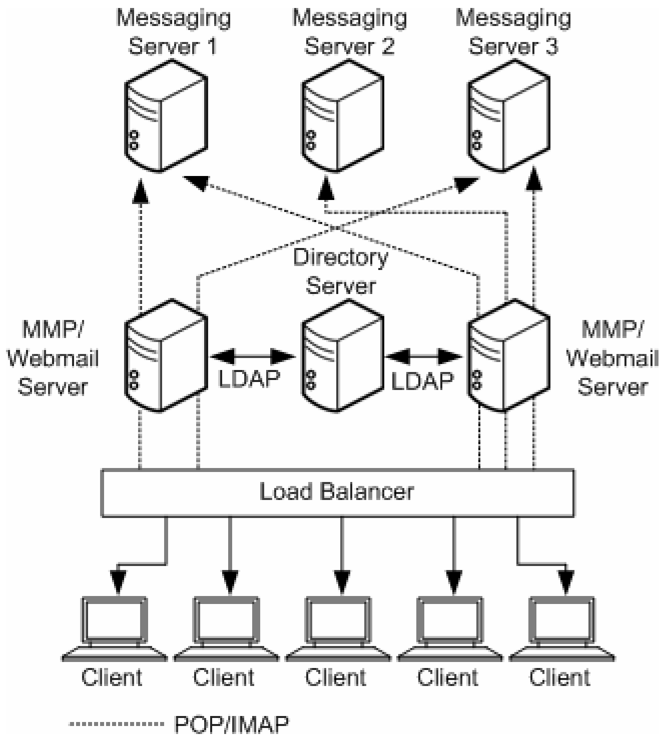
- Spreading Your Messaging User Base Across Several Servers
- Spreading Your Messaging Resources Across Redundant Components

Spreading Your Messaging User Base Across Several Servers

To distribute load across servers is to divide clients' mail evenly across several back-end Message Stores. You can divide up users alphabetically, by their Class of Service, by their department, or by their physical location and assign them to a specific back-end Message Store host. For Cassandra message store (and classic message store automatic failover), the Message Store host is replaced by Message Store affinity group. For classic message store without automatic failover, the affinity group refers to a single host or two hosts with HA failover.

Figure 3–2 shows a sample deployment where users are spread across multiple back-end servers and multiplexors enabled to handle incoming client connections.

Figure 3–2 Spreading Your User Base Across Multiple Servers



Spreading users across back-end servers provides simplified user management, as long as you use MMPs or Webmail Servers. Because users connect to one back-end server, where their mail resides, you can standardize setup across all users. This configuration also makes administration of multiple servers easier to manage. And, as the demand for more Messaging Server hosts increases, you can add more hosts seamlessly.

In Cassandra message store, and classic message store (Messaging Server 8.0.1), you can spread users across multiple affinity groups to provide horizontal scalability, allowing the deployment to service more users than it could otherwise. Affinity groups associate a set of users and machines together making internal caches more efficient and isolating notifications to just the affinity group. Routing to multi-host affinity groups is provided by MMPs, LMTP clients, and mshttpd.

Spreading Your Messaging Resources Across Redundant Components

If email is a critical part of your organization's day-to-day operations, redundant components, like load balancers, mail exchange (MX) records, and relays are necessary to ensure that the messaging system remains operational.

By using redundant MTAs, you ensure that if one component is disabled, the other is still available. Also, spreading resources across redundant MTAs enables load sharing. This redundancy also provides fault tolerance to the Messaging Server system. Each MTA relay should be able to perform the function of other MTA relays.

Installing redundant network connections to servers and MTAs also provides fault tolerance for network problems. The more critical your messaging deployment is to your organization, the more important it is for you to consider fault tolerance and redundancy.

Additional information on "Mail Exchange (MX) Records," and "Inbound and Outbound MTAs" is described in the following sections.

Mail Exchange (MX) Records

MX records are a type of DNS record that maps one host name to another. Equal priority MX records route messages to redundant inbound MTAs. For example, sending MTA from the Internet will find that the MX record for *example.com* corresponds to *MTAA.example.com* and *MTAB.example.com*. One of these MTAs is chosen at random, as they have equal priority, and an SMTP connection is opened. If the first MTA chosen does not respond, the mail goes to the other MTA. See the following MX record example:

```
example.com. in MX 10 MTAA.example.com
example.com. in MX 10 MTAB.example.com
```

Inbound and Outbound MTAs

When Messaging Server hosts are each supporting many users, and there is a heavy load of sending SMTP mail, offload the routing task from the Messaging Server hosts by using separate inbound and outbound MTAs. You can further share the load by designating different MTAs to handle outgoing and incoming messages.

Often, both the inbound and outbound MTAs are combined as a single In/Out SMTP host. To determine if you need one or more MTA hosts, identify the inbound and outbound message traffic characteristics of the overall architecture.

Load Balancers

Load balancing can be used to distribute the load across several servers so that no single server is overwhelmed. A load balancer takes requests from clients and redirects them to an available server by algorithms such as keeping track of each server's CPU and memory usage. Load balancers are available as software that runs on a common server, as a pure external hardware solution, or as a combined hardware and software package.

Planning for Vertical Scalability

Vertical scalability pertains to adding resources to individual server machines, for example, adding additional CPUs. Each machine is scaled to handle a certain load. In general, you might decide upon vertical scalability in your deployment because you have resource limitations or you are unable to purchase additional hardware as your deployment grows.

To vertically scale your deployment, you need to:

- Size each messaging component. See "Developing Messaging Server Architectural Strategies."
- Test the load of a prototype of your system. See "Using a Messaging Server Load Simulator."
- Monitor system performance and adjust the deployment accordingly.

Planning for a Highly Available Messaging Server Deployment

Note: For Cassandra message store, the Cassandra database is designed to be highly available with no single point of failure. Thus, you do not need to install additional cluster software to create a high availability solution for the message store itself. Additionally, for Cassandra message store, use of multi-host affinity groups is recommended and availability for MMP/MTA/**mshttpd** components is the same as for classic store. Also you should configure Indexed Search Converter (ISC) hosts for failover. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

High availability is a design for your deployment that operates with a small amount of planned and unplanned downtime. Typically, a highly available configuration is a cluster that is made up of two or more loosely coupled systems. Each system maintains its own processors, memory, and operating system. Storage is shared between the systems. Special software binds the systems together and allows them to provide fully automated recovery from a single point of failure. Messaging Server provides high-availability options that support the Oracle Solaris Cluster services, Oracle Clusterware, and Veritas clustering solutions.

When you create your high availability plan, you need to weigh availability against cost. Generally, the more highly available your deployment is, the more its design and operation will cost.

High availability is an insurance against the loss of data access due to application services outages or downtime. If application services become unavailable, an organization might suffer from loss of income, customers, and other opportunities. The value of high availability to an organization is directly related to the costs of downtime. The higher the cost of downtime, the easier it is to justify the additional expense of having high availability. In addition, your organization might have service level agreements guaranteeing a certain level of availability. Not meeting availability goals can have a direct financial impact.

Where to go next:

- Designing for Service Availability
- Planning a Messaging Server Sizing Strategy
- Performance Tuning Considerations for a Messaging Server Architecture

4

Planning a Messaging Server Sizing Strategy

When you design your deployment, you must decide how to configure your Oracle Communications Messaging Server to provide optimum performance, scalability, and reliability.

Sizing is an important part of this effort. The sizing process enables you to identify what hardware and software resources are needed so that you can deliver your desired level of service or response time according to the estimated workload that your Messaging Server users generate. Sizing is an iterative effort.

This chapter provides information on the basics of sizing your Messaging Server deployment to enable you to obtain the right sizing data by which you can make deployment decisions. It also provides the context and rationale for the Messaging Server sizing process.

For information about sizing the Cassandra message store, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Note: Because each deployment has its own set of unique features, this chapter does not provide detailed sizing information for your specific site. Rather, this chapter explains what you need to consider when you architect your sizing plan. Work with consulting for your deployment hardware and software needs.

Collecting Messaging Server Sizing Data

Use this section to identify the data you need to size your Messaging Server deployment. The following topics are covered in this section:

- Determining Messaging Peak Volume
- Creating Your Messaging Usage Profile
- Defining Your Messaging User Base

Determining Messaging Peak Volume

Your peak volume is the largest concentrated numbers of transactions to your messaging system within a given period in a day. The volume can vary from site to site as well as across different classes of users. For example, peak volume among a certain class of managers in a medium-sized enterprise might occur from 9 a.m. to 10 a.m. in the morning, 12 p.m. to 1 p.m. in the afternoon, and 5 p.m. to 6 p.m. in the evening.

Analyzing peak volume involves three basic operations:

1. Determining when and for how long the peaks occur.

2. Sizing your deployment against peak volume load assumptions.

   Once patterns are analyzed, choices can be made to help the system handle the load and provide the services that users demand.

3. Making sure that your Messaging Server deployment can support the peak volume that you have determined.

Creating Your Messaging Usage Profile

Measuring your load is important for accurate sizing. Your usage profile determines the factors that programs and processes place on your Messaging Server hosts.

This section helps you create your usage profile to measure the amount of load that is placed on your deployment.

To create a usage profile, answer the following questions:

1. What is the number of users on your system? When counting the number of users on your system, account for not only the users who have mail accounts and can log in to the mail system, but also the users with mail accounts who are currently not logged onto the system. In particular, note the difference between active and inactive users:

Table 4–1 Active and Inactive User Definitions

User	Description
Active User	A user who is logged into mail systems through mail access protocols like POP, IMAP, or HTTP. Depending on the type of access protocol, active users might or might not have connections to the mail server at any given time. For example, POP users can have a mail client open, but the POP connection established by the mail client to the server is short in duration and periodic. Active users in this discussion are not the same as mail attributes with **active** status, such as **mailuserstatus** or **inetuserstatus**. For more information, see the discussion on mail attributes in *Schema Reference*.
Inactive User	A user with a mail account who currently is not using the mail system.

   If you have a very small deployment (for example, under 300 users), you might not need to go through this process of planning a sizing strategy. Work with consulting to determine your individual needs.

2. How many connections are on your system during your peak volume for your POP, IMAP, and web client access services? Specifically, note the number of concurrent, idle, and busy connections for each client access service that you support:

Table 4–2 Messaging Server Connections

Connection	Description
Concurrent Connection	Number of unique TCP connections or sessions (HTTP, POP, or IMAP) that are established on your mail system at any given time. An active user can have multiple concurrent IMAP sessions, whereas a user with a POP or web client can only have one connection per client. Furthermore, because POP and web connections connect to the server, retrieve data, disconnect from the server, display data, get user input, and reconnect to the mail server, it is possible for active users on POP and web client access services not to have active connections at a given moment in time.
Idle Connection	An established IMAP connection where no information is being sent between the mail client and Messaging Server, except the occasional **check** or **noop** command.
Busy Connection	A connection that is in progress. An example of a busy connection is a mail server that is processing the command a mail client has just sent; the mail server is sending back a response to the mail client.

To determine the number of concurrent connections in your deployment, do one of the following:

- Count the number of established TCP connections by using the **netstat** command on UNIX platforms.
- Obtain the last login and logout times for web or for IMAP users. For more information, see the discussion on the **imsconnutil** command in *Messaging Server System Administrator's Guide*.

3. If you have a large deployment, how will you organize your users? Some options include but are not limited to:
   - Placing active users and inactive users together on separate machines from one another. If an inactive user becomes an active user, that user can be moved to the active user machines. This approach could decrease the amount of needed hardware, rather than placing inactive and active users together on a machine.
   - Separating users by Class of Service. You might separate individual contributors, managers, and executives on machines that offer different mail storage space allocation for each class of service, different privileges, and specialized services.
4. What is the amount of storage used on each mailbox? When you measure the amount of storage per mailbox, you should estimate real usage per mailbox, not the specified quota. Messages in trash or wastebasket folders still take up disk space and quota.
5. How many messages enter your messaging system from the Internet? The number of messages should be measured in messages per second during your peak volume.
6. How many messages are sent by your users to:
   - End users on your mail system?
   - The Internet? This number of messages is also measured in messages per second during the peak volume.
7. What is the distribution of messages in different size ranges? For example:

- Less than 5 Kbytes?
- Between 5 Kbytes - 10 Kbytes?
- Between 10 Kbytes - 100 Kbytes?
- Between 100 Kbytes - 500 Kbytes?
- Between 500 Kbytes - 10 MB?
- Greater than 10 MB? If the distribution of message sizes is not available, use the average message size on your mail system, however it is not as effective as size ranges. The size of messages is particularly important, because it affects the rate of delivery of the MTA, the rate of delivery into the Message Store, the rate of message retrieval, and processing by anti-virus or anti-spam filters.

8. Will you be using SSL/TLS? If yes, what percentage of users and what type of users? For example, in a particular organization, 20 percent of IMAP connections during peak hours will enable SSL.
9. Do you plan on using any SSL crypto accelerator hardware?
10. Will you be using virus scanning or other specialized message processing and will this processing be enabled for all users? Depending on your Messaging Server configuration, the MTA will need to scan all messages to match criteria specified in specialized processing, thus increasing load on the system.
11. For POP users, will you have a policy restricting how often they can access mail? If so, how often?
12. For IMAP users, will you enforce a standard client or allow users to choose their own? Different IMAP clients make different numbers of concurrent connections to the server. Thus, a power user with many open folders might have many concurrent connections.
13. Will you allow users to share folders? If so, will you allow all users or only some?

Answering these questions provides a preliminary usage profile for your deployment. You can refine your usage profile as your Messaging Server needs change.

Additional Questions

While the following questions are not applicable to creating your usage profile, they are important to developing your sizing strategy. How you answer these questions might require you to consider additional hardware.

1. How much redundancy do you want in your deployment?
2. What backup and restore strategy do you have in place (such as disaster recovery, mailbox restores, and site failover)? What are the expected times to accomplish recovery tasks?
3. Do you need a DMZ to separate your internal and external networks? Are all users using the internal network? Or do some of them connect by using the Internet? You might need MMP proxy servers and separate MTA layers.
4. What are your response time requirements? What are your throughput requirements?
5. What is your specific criteria for resource utilization? Can your CPUs be 80 percent busy on average? Or only at peak?
6. Will you have messaging servers at different geographic locations? Do you expect users' mail to be located geographically?

7. Do you have archiving requirements for keeping mail messages for a certain length of time?
8. Do you have legal requirements to log all messages? Do you need to keep a copy of every message sent and received?

Defining Your Messaging User Base

Once you establish a usage profile, compare it to sample pre-defined user bases that are described in this section. A **user base** is made up of the types of messaging operations that your users will perform along with a range of message sizes that your users will send and receive. Messaging users fall into one of five user bases:

- Lightweight POP
- Heavyweight POP
- Lightweight IMAP
- Mediumweight IMAP
- Mediumweight Convergence

The sample user bases described in this section broadly generalize user behavior. Your particular usage profile might not exactly match the user bases. You will be able to adjust these differences when you run your load simulator (as described in "Using a Messaging Server Load Simulator").

Lightweight POP

A lightweight POP user base typically consists of residential dial-up users with simple messaging requirements. Each concurrent client connection sends approximately four messages per hour. These users read and delete all of their messages within a single login session. In addition, these users compose and send few messages of their own with just single recipients. Approximately 80 percent of messages are 5 Kbytes or smaller in size, and about 20 percent of messages are 10 Kbytes or larger.

Heavyweight POP

A heavyweight POP user base typically consists of premium broadband users or small business accounts with more sophisticated messaging requirements than the lightweight POP user base. This group uses cable modem or DSL to access its service provider. Each concurrent client connection sends approximately six messages per hour. Messages average about two recipients per message. Sixty-five percent of messages are 5 Kbytes or smaller in size. Thirty percent of messages in this user base are between 5-10 Kbytes. Five percent of messages are larger than 1 Mbyte. Of these users, 85 percent delete all of their messages after reading them. However, 15 percent of users leave messages on the server through several logins before they delete them. Mail builds up in a small portion of those mailboxes. In some cases, the same message can be fetched several times from the server.

Lightweight IMAP

A lightweight IMAP user base represents users that enable premium broadband Internet services, including most of the advanced features of their messaging systems like message searching and client filters. This user base is similar to heavyweight POP with regard to message sizes, number of recipients, and number of messages sent and received by each concurrent connection. Lightweight IMAP users typically log in for hours at a time and delete most or all mail before log out. Consequently, mail stacks up in a mailbox during a login session, but user generally do not store more than 20 to 30 messages in their mailboxes. Most inboxes contain less than 10 messages.

Mediumweight IMAP

A mediumweight IMAP user base represents sophisticated enterprise users with login sessions lasting most of an eight hour business day. These users send, receive, and keep a large amount of mail. Furthermore, these users have unlimited or very large message quotas. Their inboxes contain a large amount of mail that grows during the day, and is fully or partially purged in large spurts. They regularly file messages into folders and search for messages multiple times per hour. Each concurrent client connection sends approximately eight messages per hour. These users send messages with an average of four recipients and have the same message size mix as the Heavyweight POP and Lightweight IMAP user bases.

Mediumweight Convergence

A mediumweight Convergence user base is similar to Mediumweight IMAP. This user base has the same message size mix as Mediumweight IMAP, Lightweight IMAP, and Heavyweight POP. And, the message delivery rates are the same as Mediumweight IMAP users.

It is likely that you will have more than one type of user base in your organization, particularly if you offer more than one client access option. Once you identify your user bases from these categories, you will test them with your usage profile and with a load simulator, described in "Using a Messaging Server Load Simulator".

Using a Messaging Server Load Simulator

To measure the performance of your Messaging Server, use your messaging user base (described in "Defining Your Messaging User Base") and your messaging usage profile (described in "Creating Your Messaging Usage Profile") as inputs into a load simulator.

A load simulator creates a peak volume environment and calibrates the amount of load placed on your servers. You can determine if you need to alter your hardware, throughput, or deployment architecture to meet your expected response time, without overloading your system.

To Use a Load Simulator

1. Define the user base that you want to test (for example, Lightweight IMAP). If necessary, adjust individual parameters to best match your usage profile.
2. Define the hardware that will be tested.
3. Run the load simulator and measure the maximum number of concurrent connections on the tested hardware with the user base.
4. Publish your results and compare those results with production deployments.
5. Repeat this process using different user bases and hardware until you get the response time that is within an acceptable range for your organization under peak load conditions.

Note: Contact consulting for recommended load simulators and support.

Assessing Your Messaging Server System Performance

Once you evaluate your hardware and user base with a load simulator, you need to assess your system performance. The following topics address methods by which you can improve your overall system performance.

Messaging Server Memory Utilization

Make sure you have an adequate amount of physical memory on each machine in your deployment. Additional physical memory improves performance and enables the server to operate at peak volume. Without sufficient memory, Messaging Server cannot operate efficiently without excessive swapping.

At minimum, be sure to have 1 GB of memory per CPU. For most deployments, you will want 2 GB of memory per CPU.

Messaging Server Disk Throughput

Note: With Cassandra message store, the disk throughput is provided by SSDs on Cassandra store nodes. Use of faster SSDs can improve throughput, but use of more Cassandra hosts also improves performance and is likely the simpler approach to improve throughput to meet requirements.

Disk throughput is the amount of data that your system can transfer from memory to disk and from disk to memory. The rate at which this data can be transferred is critical to the performance of Messaging Server. To create efficiencies in your system's disk throughput:

- Consider your maintenance operations, and ensure you have enough bandwidth for backup. Backup can also affect network bandwidth particularly with remote backups. Private backup networks might be a more efficient alternative.
- Carefully partition the store and separate store data items (such as **tmp** and **db**) to improve throughput efficiency.
- Ensure the user base is distributed across RAID (Redundant Array of Independent Disks) environments in large deployments.
- Stripe data across multiple disk spindles in order to speed up operations that retrieve data from disk.
- Allocate enough CPU resources for RAID support, if RAID does not exist on your hardware.

You want to measure disk I/O in terms of IOPS (total I/O operations per second) not bandwidth. You need to measure the number of unique disk transactions the system can handle with a very low response time (less than 10 milliseconds).

Messaging Server Disk Capacity

When planning server system disk space, you need to be sure to include space for operating environment software, Messaging Server software, and message content and tracking. For classic message store, be sure to use an external disk array if availability is a requirement. For most systems, external disks are required for performance because the internal system disks supply no more than four spindles.

For the classic message store partitions, the storage requirement is the total size of all messages plus 30 percent overhead.

In addition, user disk space needs to be allocated. Typically, this space is determined by your site's policy.

Note: Your deployment planning needs to include how you want to back up the classic message store for disaster recovery.

Disk Sizing for MTA Message Queues

The behavior of the Messaging Server MTA Queue is to provide a transient store for messages waiting to be delivered. Messages are written to disk in a persistent manner to maintain guaranteed service delivery. If the MTA is unable to deliver the message, it will retry until it finally gives up and returns the message to the sender.

Message Queue Availability

SMTP services are considered a guaranteed message delivery service. This is an assurance to end users that the Messaging Server will not lose messages that the service is attempting to deliver. When you architect the design of the MTA Queue system, all effort should be made to ensure that messages will not be lost. This guarantee is usually made by implementing redundant disk systems through various RAID technologies.

Message Queue Available Disk

The queue will grow excessively if one of the following conditions occurs:

- The site has excessive network connectivity issues
- The MTA configuration is holding on to messages too long
- There are valid problems with those messages (not covered in this document)

The following sections address these issues.

Planning for Network Connectivity Issues

Occasionally the MTA is unable to deliver messages due to network connectivity issues. In these cases, the messages will be stored on the queue until the next time the MTA is able to attempt to deliver (as defined by the retry interval).

Planning on disk space for these outages is based on a simple rule, the "General Rule for Message Queue Sizing:"

1. Determine average number of messages/minute expected to be delivered (N).
2. Determine average size (kb) of messages (S).
3. Determine maximum duration (minutes) of typical network connectivity outages (T).

Thus, the formula for estimating the Disk Queue Size is:

Disk Queue Size (kb) = N x S x T

Tuning MTA for Reattempts of Delivery

Occasionally, the system will not be able to deliver any messages. In this state, messages will reside on the message queue while the MTA attempts to set aside the messages for a period of time (retry interval) until it reattempts the delivery. This will continue until the MTA gives up and returns the message to the sender. The reason a message is undeliverable is fairly unpredictable. A number of reasons such as network connectivity, busy destination server, network throttles, and so on, could explain why the message is undeliverable.

On a busy server, these temporarily stored messages can build up during periods of high volume activities. Such a build-up can potentially cause problems with disk space. To avoid these build-ups, tune the MTA to retry delivery at a faster rate.

The retry interval is set within the Channel Block configurations of the **imta.cnf** file. The structure of this file consists of two parts: rewrite rules and channel blocks. The channel blocks define the behavior of a particular disk queue and related processes. This discussion refers to the **tcp_local** channel. The **tcp_local** channel provides delivery to sites outside an enterprise's local network, in other words, to places over the Internet.

The retry interval setting of the **tcp_local** channel is initially set by the default channel block. The default channel block allows settings to be duplicated to avoid having repeated settings.

The following is the default channel block:

```
defaults notices 1 2 4 7 copywarnpost copysendpost postheadonly
noswitchchannel immnonurgent maxjobs 7 defaulthost
red.example.com red.example.com
```

First, the structure of the channel block consists of the channel name. In the example above, this is the default channel block, which will be applied to channels without these settings. The second part is a list of channel keywords.

The **notices** keyword specifies the amount of time that can elapse before message delivery notices (MDNs) are sent back to the sender. This keyword starts with the **notices** keyword followed by a set of numbers, which set the retry period. By default, the MTA will attempt delivery and send notices back to the sender. These notices come from “postmaster” to end-user inboxes.

In this example, the MTA will retry at a period of 1 day, 2 days, and 4 days. At 7 days, the MTA will return the message and regard the message as a failed delivery.

In many cases, the default setting of the MTA provides adequate performance. In some cases, you need to tune the MTA to avoid potential resource exhaustions, such as running out disk space for message queues. This is not a product limitation, but a limitation of the total Messaging Server system, which includes hardware and network resources.

In consideration of these possible disk size issues, deployments with a large number of users may not want to attempt message deliveries for much shorter intervals. If this is the case, study refer to the following documentation for more information.

- See the discussion on how to set notification message delivery in *Messaging Server System Administrator's Guide*.
- See the discussion on configuring channel definitions in *Messaging Server System Administrator's Guide*.

Messaging Server Network Throughput

Network throughput is the amount of data at a given time that can travel through your network between your client application and server. When a networked server is unable to respond to a client request, the client typically retransmits the request a number of times. Each retransmission introduces additional system overhead and generates more network traffic.

You can reduce the number of retransmissions by improving data integrity, system performance, and network congestion:

- To avoid bottlenecks, ensure that the network infrastructure can handle the load.
- Partition your network. For example, use 100 Mbps Ethernet for client access and 1 GB Ethernet for the backbone.
- To ensure that sufficient capacity exists for future expansion, do not use theoretical maximum values when configuring your network.
- Separate traffic flows on different network partitions to reduce collisions and to optimize bandwidth use.

Messaging Server CPU Resources

Enable enough CPU for your Message Stores, MTAs, and on systems that are just running multiplexing services (MMP). In addition, enable enough CPU for any RAID systems that you plan to use.

Performance Tuning Considerations for a Messaging Server Architecture

This information describes how to evaluate the performance characteristics of Messaging Server components to accurately develop your architecture and ensure proper tuning of the deployment.

Message Store Performance Considerations

Note: Starting with the release of Messaging Server 8.0.2, you can choose to use either classic message store or "Cassandra" message store, which is based on DataStax Cassandra Database. You cannot use both message stores in the same deployment. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information about installing Cassandra message store.

Message Store performance is affected by a variety of factors, including:

- Disk I/O
- Inbound message rate (also known as message insertion rate)
- Message sizes
- Login rate (POP/IMAP/HTTP)
- Transaction rate for IMAP and HTTP
- Concurrent number of connections for the various protocols
- Network I/O
- Use of SSL

The preceding factors list the approximate order of impact to the Message Store. Most performance issues with the Message Storage arise from insufficient disk I/O capacity. Additionally, the way in which you lay out the store on the physical disks can also have a performance impact. For smaller standalone systems, it is possible to use a simple stripe of disks to provide sufficient I/O. For most larger systems, segregate the file system and provide I/O to the various parts of store.

In addition to tuning the Message Store, you need to protect the Message Store from loss of data. The level of loss and continuous availability that is necessary varies from simple disk protection such as RAID5 (classic message store) or Cassandra replication factor, to mirroring, to routine backup, to real time replication of data, to a remote data center. Data protection also varies from the need for Automatic System Recovery (ASR) capable machines, to local HA capabilities, to automated remote site failover. These decisions impact the amount of hardware and support staff required to provide service.

Messaging Server Directories

Messaging Server uses six directories that receive a significant amount of input and output activity. If you require a deployment that is scalable, responsive, and resilient to variations in load, provide each of those directories with sufficient I/O bandwidth. When you provide separate file systems for these directories, each composed of multiple drives, you can more readily diagnose I/O bottlenecks and problems. Also, you can isolate the effect of storage failures and simplify the resulting recovery operations. In addition, place a seventh directory for DB snapshots on a file system separate from the active DB to preserve it in the event of a storage failure of the active DB file system.

Table 4–3 describes these directories.

Table 4–3 High Access Messaging Server Directories

High I/O Directory	Description and Defining Options
MTA queue directory	In this directory, many files are created, one for each message that passes through each MTA channel. After the file is sent to the next destination, the file is then deleted. The directory is located at *DataRoot*/**queue**. After moving it to another file system, replace *DataRoot*/**queue** with a softlink. Also see the **subdirs** channel option.
Messaging Server log directory	This directory contains log files which are constantly being appended with new logging information. The number of changes will depend on the logging level set. The directory location is controlled by the **msconfig** option ***.logfile.logdir** (Unified Configuration) or the **configutil** option **logfile.*.logdir** (legacy configuration), where * can be a log-generating component such as admin, default, HTTP, IMAP, or POP. To change the location of the MTA log files, replace *DataRoot*/**log** with a softlink.
Mailbox database files	These files require constant updates as well as cache synchronization. Put this directory on your fastest disk volume. These files are always located in the DataRoot**/store/mboxlist** directory.
Message store index files	These files contain meta information about mailboxes, messages, and users. By default, these files are stored with the message files. The **msconfig** option **partition:*.path** (Unified Configuration) or **configutil** option **store.partition.*.path** (legacy configuration), where * is the name of the partition, controls the directory location. If you have the resources, put these files on your second fastest disk volume. Default location (classic message store only): *DataRoot*/**store/partition/primary**

Table 4–3 (Cont.) High Access Messaging Server Directories

High I/O Directory	Description and Defining Options
Message files	These files contain the messages, one file per message. Files are frequently created, never modified, and eventually deleted. By default, they are stored in the same directory as the message store index files. The location can be controlled with the **msconfig** option **partition:***.**messagepath** (Unified Configuration) or the **configutil** option **store.partition.*.messagepath** (legacy configuration), where * is the name of the partition. Some sites might have a single message store partition called **primary** specified by **partition:primary.path** (Unified Configuration) or **store.partition.primary.path** (legacy configuration). Large sites might have additional partitions that can be specified with **store.partition.***partition_name***.messagepath**, where *partition_name* is the name of the partition. Default location (classic message store only): *DataRoot***/store/partition/primary**
Mailbox list database temporary directory	The directory used by the Message Store for database temporary files. To maximize performance, this directory should be located under the fastest file system. The default value is **/tmp/.***ENCODED_SERVERROOT***/store/**, where *ENCODED_SERVERROOT* is composed of the mail server user plus the value of **$SERVERROOT** with / replaced by _.
Lock directory	DB temporary files used for locking other resources. Use a **tmpfs** directory similar to **store.dbtmpdir**, but do NOT use the same directory. For example, **msconfig** uses **base.lockdir** (Unified Configuration), or **configutil -o local.lockdir -v /tmp/msgDBlockdir** for legacy configuration. Be sure to use a unique name so the same directory cannot be used by any other instance of Messaging Server. Default location: **/tmp/***.ENCODED_SERVERROOT***/lock/** on Solaris and **/dev/shm/***.ENCODED_SERVERROOT***/lock/** on Linux, where *ENCODED_SERVERROOT* is composed of mail server user plus the **$SERVERROOT** with / replaced by _.

The following sections provide more detail on Messaging Server high access directories.

MTA Queue Directories

Because the recommended configuration for Messaging Server is to use LMTP to deliver messages, the message store server runs only the LMTP server and does not run an MTA. Use LMTP for both classic message store and Cassandra message store.

LMTP does not use MTA message queues at all. It works such that inbound messages are not put in MTA queues but directly inserted into the store. This message insertion lessens the overall I/O requirements of the Message Store machines and greatly reduces use of the MTA queue directory on Message Store machines. In a two-tiered environment using LMTP, this directory will be lightly used, if at all. In prior releases of Messaging Server, on large systems this directory set needs to be on its own stripe or volume.

When an MTA is co-located with IMAP or POPD processes, then queues are present and must be managed and monitored.

In non-LMTP environments, the MTA queue directories in the Message Store system are heavily used. MTA queue directories should usually be on their own file systems, separate from the message files in the Message Store. The Message Store has a mechanism to stop delivery and appending of messages if the disk space drops below a defined threshold. However, if both the log and queue directories are on the same file system and keep growing, you will run out of disk space and the Message Store will stop working.

Also, refer to the **subdirs** channel option. If a channel will often contain many messages, it may be necessary to increase the number of subdirectories for that channel queue directory.

Log Files Directory

The log files directory requires varying amounts of I/O depending on the level of logging that is enabled. The I/O on the logging directory, unlike all of the other high I/O requirements of the Message Store, is asynchronous. For typical deployment scenarios, do not dedicate an entire Logical Unit Number (LUN) for logging. For very large store deployments, or environments where significant logging is required, a dedicated LUN is in order.

mboxlist Directory

Note: This section applies only to classic message store.

The **mboxlist** directory is highly I/O intensive but not very large. The **mboxlist** directory contains the databases that are used by the stores and their transaction logs. Because of its high I/O activity, and due to the fact that the multiple files that constitute the database cannot be split between different file systems, you should place the **mboxlist** directory on its own stripe or volume in large deployments. This is also the most likely cause of a loss of vertical scalability, as many procedures of the Message Store access the databases. For highly active systems, this can be a bottleneck. Bottlenecks in the I/O performance of the **mboxlist** directory decrease not only the raw performance and response time of the store but also impact the vertical scalability. For systems with a requirement for fast recovery from backup, place this directory on Solid State Disks (SSD) or a high performance caching array to accept the high write rate that an ongoing restore with a live service will place on the file system.

Multiple Store Partitions

Note: This section applies only to classic message store.

The Message Store supports multiple store partitions. Place each partition on its own stripe or volume. The number of partitions that should be put on a store is determined by a number of factors. The obvious factor is the I/O requirements of the peak load on the server. By adding additional file systems as additional store partitions, you increase the available IOPS (total IOs per second) to the server for mail delivery and retrieval. In most environments, you will get more IOPS out of a larger number of smaller stripes or LUNs than a small number of larger stripes or LUNs.

With some disk arrays, it is possible to configure a set of arrays in two different ways. You can configure each array as a LUN and mount it as a file system. Or, you can configure each array as a LUN and stripe them on the server. Both are valid configurations. However, multiple store partitions (one per small array or a number of partitions on a large array striping sets of LUNs into server volumes) are easier to optimize and administer.

Raw performance, however, is usually not the overriding factor in deciding how many store partitions you want or need. In corporate environments, it is likely that you will need more space than IOPS. Again, it is possible to software stripe across LUNs and provide a single large store partition. However, multiple smaller partitions are generally easier to manage. The overriding factor of determining the appropriate number of store partitions is usually recovery time.

Recovery times for store partitions fall into a number of categories:

- First of all, the **fsck** command can operate on multiple file systems in parallel on a crash recovery caused by power, hardware, or operating system failure. If you are using a journaling file system (highly recommended and required for any HA platform), this factor is small.
- Secondly, backup and recovery procedures can be run in parallel across multiple store partitions. This parallelization is limited by the vertical scalability of the **mboxlist** directory as the Message Store uses a single set of databases for all of the store partitions. Store cleanup procedures (**expire** and **purge**) run in parallel with one thread of execution per store partition.
- Lastly, mirror or RAID re-sync procedures are faster with smaller LUNs. There are no hard and fast rules here, but the general recommendation in most cases is that a store partition should not encompass more than 10 spindles.

The size of drive to use in a storage array is a question of the IOPS requirements versus the space requirements. For most residential ISP POP environments, use “smaller drives.” Corporate deployments with large quotas should use “larger” drives. Again, every deployment is different and needs to examine its own set of requirements.

Message Store Processor Scalability

Note: This section applies only to classic message store.

The Message Store scales well, due to its multiprocess, multithreaded nature. The Message Store actually scales more than linearly from one to four processors. This means that a four processor system will handle more load than a set of four single processor systems. The Message Store also scales fairly linearly from four to 12 processors. From 12 to 16 processors, there is increased capacity but not a linear increase. The vertical scalability of a Message Store is more limited with the use of LMTP although the number of users that can be supported on the same size store system increases dramatically.

Setting the Mailbox Database Cache Size

Note: This section applies only to classic message store.

Messaging Server makes frequent calls to the mailbox database. For this reason, it helps if this data is returned as quickly as possible. A portion of the mailbox database is cached to improve Message Store performance. Setting the optimal cache size can make a big difference in overall Message Store performance. You set the size of the cache with the **store.dbcachesize** option.

The **store.dbcachesize** option defaults to **/tmp/.***ENCODED_SERVERROOT***/store/**, where *ENCODED_SERVERROOT* is composed of the mail server user plus the value of **$SERVERROOT** with the backslash (\) replaced by _. For example: **/tmp/.mailsrv_ opt_sun_comms_messaging64/store/**

The files stored in the **store.dbtmpdir** location are temporarily memory mapped files used by all processes connecting to the database. Due to their usage pattern, the pages of these files will most likely be in memory all the time. So setting this to be on a **tempfs** will not really increase memory usage. What it will do is save I/O. When the Oracle Solaris virtual memory system sees a memory mapped file is on a **tempfs**, it knows it does not really need to write the modified pages back to the file. So there is only one copy in memory and it saves I/O.

The mailbox database is stored in data pages. When the various daemons make calls to the database (**stored**, **imapd**, **popd**), the system checks to see if the desired page is stored in the cache. If it is, the data is passed to the daemon. If not, the system must write one page from the cache back to disk, and read the desired page and write it in the cache. Lowering the number of disk read/writes helps performance, so setting the cache to its optimal size is important.

If the cache is too small, the desired data will have to be retrieved from disk more frequently than necessary. If the cache is too large, dynamic memory (RAM) is wasted, and it takes longer to synchronize the disk to the cache. Of these two situations, a cache that is too small will degrade performance more than a cache that is too large.

Cache efficiency is measured by hit rate. Hit rate is the percentage of times that a database call can be handled by cache. An optimally sized cache will have a 98 to 99 percent hit rate (that is, 98 to 99 percent of the desired database pages will be returned to the daemon without having to grab pages from the disk). The goal is to set the smallest cache so that it holds a number of pages such that the cache will be able to return at least 98 to 99 percent of the requested data. If the direct cache return is less than 98 percent, then you need to increase the cache size.

To Adjust the Mailbox Database Cache Size

Note: This section applies only to classic message store.

Set the size of the cache with the **msconfig** option (Unified Configuration) or **configutil** option (legacy configuration) to **store.dbcachesize**.

It is important to tune the cache size to smallest size that will accomplish the desired hit rate.

The **store.dbcachesize** controls the size of a shared memory segment used by all processes connected to the database, including **stored**, **imap**, **popd**, **imsbackup**, **imsrestore**, **ims_master**, **tcp_lmtp_server**, and so on. While the maximum value for store.dbcachesize is 2 GB, setting it to the maximum consumes half of the 32-bit address space of your those processes. Instead, start with the default value of 16 MB and monitor the cache hit rate over a period of days. Increase the value only if the hit rate is under 98 percent.

Also consider the transaction checkpoint function (performed by **stored**). Set the **msconfig** option (Unified Configuration) or **configutil** option (legacy configuration) to **store.checkpoint.debug** and refresh **stored** to see log messages to provide more exact data about transaction checkpoint function time. This process must examine all buffers in the cache and hold a region lock during the checkpoint. Other threads needing the lock must wait.

To Monitor the Mailbox Database Cache Size

Note: This section applies only to classic message store.

Use the **imcheck** command to measure the cache hit rate:

```
imcheck -s mpool > imcheck-s.out
```

In this example, Messaging Server is installed in **/opt/sun/comms/messaging64** and **store.dbtmpdir** is set to **/tmp/msgDBtmpdir**.

Find the cache information section in the output file, for example:

```
2MB 513KB 604B Total cache size.
1 Number of caches.
1       Maximum number of caches
2MB 520KB Pool individual cache size.
```

There will be several blocks of output, including a summary and one for each database file. Look for these lines in each block:

```
0 Requested pages mapped into the process' address space.
55339 Requested pages found in the cache (99%).
```

In this case, the hit rate is 99 percent. This could be optimal or, more likely, it could be that the cache is too large. To test, lower the cache size until the hit rate moves to below 99 percent. When you hit 98 percent, you have optimized the DB cache size. Conversely, if see a hit rate of less than 95 percent, then you should increase the cache size with the **store.dbcachesize** option.

As your user base changes, the hit rate can also change. Periodically check and adjust this option as necessary.

Setting Disk Stripe Width

When setting disk striping, the stripe width should be about the same size as the average message passing through your system. A stripe width of 128 blocks is usually too large and has a negative performance impact. Instead, use values of 8, 16, or 32 blocks (4, 8, or 16 kilobyte message respectively).

MTA Performance Considerations

MTA performance is affected by a number of factors including, but not limited to:

- Spam and virus filtering
- Disk performance
- Use of SSL
- The number of messages/connections inbound and outbound
- The size of messages
- The number of target destinations/messages
- The speed and latency of connections to and from the MTA
- The use of Sieve rules and the need to do other message parsing (like use of the conversion channel)

The MTA is both CPU and I/O intensive. The MTA reads from and writes to two different directories: the queue directory and the logging directory. For a small host (four processors or less) functioning as an MTA, you do not need to separate these directories on different file systems. The queue directory is written to synchronously with fairly large writes. The logging directory is a series of smaller asynchronous and sequential writes. On systems that experience high traffic, consider separating these two directories onto two different file systems.

In most cases, you will want to plan for redundancy in the MTA in the disk subsystem to avoid permanent loss of mail in the event of a spindle failure. (A spindle failure is by far the single most likely hardware failure.) This implies that either an external disk array or a system with many internal spindles is optimal.

MTA and Raid Trade-offs

There are trade-offs between using external hardware RAID controller devices and using JBOD arrays with software mirroring. The JBOD approach is sometimes less expensive in terms of hardware purchase but always requires more rack space and power. The JBOD approach also marginally decreases server performance, because of the cost of doing the mirroring in software, and usually implies a higher maintenance cost. Software RAID5 has such an impact on performance that it is not a viable alternative. For these reasons, use RAID5 caching controller arrays if RAID5 is preferred.

MTA and Processor Scalability

The MTA does scale linearly beyond eight processors, and like the Message Store, more than linearly from one processor to four.

MTA and High Availability

It is rarely advisable to put the MTA under HA control, but there are exceptional circumstances where this is warranted. If you have a requirement that mail delivery happens in a short, specified time frame, even in the event of hardware failure, then the MTA must be put under HA software control. In most environments, simply increase the number of MTAs that are available by one or more over the peak load requirement. This ensures that proper traffic flow can occur even with a single MTA failure, or in very large environments, when multiple MTAs are offline for some reason.

In addition, with respect to placement of MTAs, you should always deploy the MTA inside your firewall.

MMP Performance Considerations

The MMP runs as a single multithreaded process and is CPU and network bound. It uses disk resources only for logging. The MMP scales most efficiently on two processor machines, scales less than linearly from two to four processors and scales poorly beyond four processors. Two processor, rack mounted machines are good candidates for MMPs.

MMP sizing is affected by connection rates and transaction rates. POP sizing is fairly straight forward, as POP connections are rarely idle. POP connections connect, do some work, and disconnect. IMAP sizing is more complex, as you need to understand the login rate, the concurrency rate, and the way in which the connections are busy. The MMP is also somewhat affected by connection latency and bandwidth. Thus, in a dial up environment, the MMP will handle a smaller number of concurrent users than in a broadband environment, as the MMP acts as a buffer for data coming from the Message Store to the client.

MMP and Webmail Server

You can put the MMP and Webmail Server on the same set of servers. The advantage of doing so is if a small number of either MMPs or Webmail Servers is required, the amount of extra hardware for redundancy is minimized. The only possible downside to co-locating the MMP and Webmail Server on the same set of servers is that a denial of service attack on one protocol can impact the others.

File System Performance Considerations

For a small but perceptible performance gain, you should enable **noatime** on your Messaging Server file systems. By default, the file system is mounted with normal access time (**atime**) recording. If you specify **noatime**, then the file system ignores the access time updates on files, reducing disk activity.

To enable **noatime**, edit the **/ect/vfstab** file's options field, for example:

```
/dev/dsk/c1d0s0 /dev/rdsk/c1d0s0 / ufs 1 no noatime
```

ZFS also has **atime** on by default as well, so you should change that to **off**. Use the **zfs set** command, for example:

```
zfs set atime=off tank/home
```

CPU Considerations

- For sites which use IMAP heavily, set **service.imap.numprocesses** to the number of CPUs (or cores on CMT systems) divided by 4.
- For POP sites, set **service.pop.numprocesses** to the number of CPUs (or cores on CMT systems) divided by 2.

Performance Tuning Realtime BlockLists (RBL) Lookups

The **dns_verify.so** Messaging Server plugin provides a mechanism to block emails based on DNS Realtime Blocklists (RBL) data. RBL Blocklists provided by organizations such as Spamhaus (see `http://www.spamhaus.org/`) provide an excellent mechanism to reduce the number of emails that are sent from IP addresses of hosts that are known or highly-likely to send spam or bulk unsolicited emails.

This section contains the following topics:

- Performance Discussion
- Hints and Tips

Performance Discussion

The use of DNS RBL lookups to reduce spam email comes at the cost of some additional CPU and network utilisation plus increased time to accept email messages due to DNS resolution delays.

The additional CPU and network utilisation tends to be negated by the overall reduction in email processing due to less spam emails – and therefore less overall emails. The increased time to accept email messages due to DNS resolution delays is a very-real issue that results in a bottleneck in the rate that emails can be accepted.

The most efficient point to see if the IP address of the connecting host is listed in a DNS Realtime Blocklists is at the initial connection state. The PORT_ACCESS mapping table is the first table that is checked, and therefore this is the table most commonly used to perform the dns_verify.so library callout.

As the dispatcher uses a single-thread-per-listen-port, the rate at which an initial email connection can be accepted, compared against the PORT_ACCESS mapping table and then handed off to the multi-threaded tcp_smtp_server process will depend on the time taken for the PORT_ACCESS mapping table comparison to be performed.

Large DNS resolution times in the dns_verify.so callout will therefore cause a bottleneck in the rate connections can be accepted and handed off. The common symptom of this bottleneck is for a system to take a long time to return the initial SMTP banner when the system is either under heavy client connection load or experiencing large DNS resolution times.

Hints and Tips

This section discusses the following topics:

- Reduce DNS Lookups
- Improve Performance of DNS Lookups

Reduce DNS Lookups

Prevention is better then cure. Careful rearrangement and modification of mapping table rules can assist in reducing the overall number of DNS lookups that are performed and therefore improve the rate that emails can be accepted.

- **Use absolute DNS lookups by adding a "." to the end of the domain**

Using a relative domain lookup e.g. *zen.spamhaus.org* vs. an absolute lookup e.g. *zen.spamhaus.org*. will result in unnecessary lookups. The number of additional lookups will depend on the systems **/etc/resolv.conf** configuration. A configuration with numerous 'search' domains defined will result in an equivalent number of additional lookups.

(relative domain lookup - a single search domain defined: aus.sun.com)

```
TCP|*|25|*|* $C$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E

mailserver.aus.sun.com -> dns.Aus.Sun.COM DNS C 3.100.168.192.zen.spamhaus.org.
Internet TXT ?
dns.Aus.Sun.COM -> mailserver.aus.sun.com DNS R  Error: 3(Name Error)
mailserver.aus.sun.com -> dns.Aus.Sun.COM DNS C 3.100.168.192.zen.spamhaus.org.
Internet Addr ?
dns.Aus.Sun.COM -> mailserver.aus.sun.com DNS R  Error: 3(Name Error)
mailserver.aus.sun.com -> dns.Aus.Sun.COM DNS C
3.100.168.192.zen.spamhaus.org.aus.sun.com. Internet Addr ?
dns.Aus.Sun.COM -> mailserver.aus.sun.com DNS R  Error: 3(Name Error)
```

(absolute domain lookup - one less lookup compared to relative domain lookup)

```
 TCP|*|25|*|* $C$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E

mailserver.aus.sun.com -> dns.Aus.Sun.COM DNS C 3.100.168.192.zen.spamhaus.org.
Internet TXT ?
dns.Aus.Sun.COM -> mailserver.aus.sun.com DNS R  Error: 3(Name Error)
```

```
mailserver.aus.sun.com -> dns.Aus.Sun.COM DNS C 3.100.168.192.zen.spamhaus.org.
Internet Addr ?
dns.Aus.Sun.COM -> mailserver.aus.sun.com DNS R  Error: 3(Name Error)
```

- **Restrict the rule to port 25 and non-internal IP addresses (after the INTERNAL_ IP)**

To avoid unnecessary lookups for internal systems, place the RBL DNS lookup rule after the default INTERNAL_IP PORT_ACCESS rule and restrict the rule to port 25 only as this prevents internal systems from being accidentally blocked and stops email submission (port 587/465) from being checked e.g.

```
PORT_ACCESS

! TCP|server-address|server-port|client-address|client-port
  *|*|*|*|*  $C$|INTERNAL_IP;$3|$Y$E
  TCP|*|25|*|* $C$:S$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
  *  $YEXTERNAL
```

- **Use the appropriate mapping table modifier for your version of Messaging Server**

If you have MS6.3 patch **120228-24** or below:

Use **$:A** to halve the number of lookups e.g.

```
 TCP|*|25|*|* $C$:A$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
```

If you have MS6.3 patch 1**20228-25 and above**:

Use **$:S** to move lookups to the multi-threaded smtp-server process e.g.

```
 TCP|*|25|*|* $C$:S$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
```

(**Not** using **$:S** or **$:A** modifier - twice the number of lookups)

```
02:30:15.629216 IP mailserver.Aus.Sun.COM.41249 > dns.Aus.Sun.COM.domain: 27201+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.629222 IP mailserver.Aus.Sun.COM.41249 > dns.Aus.Sun.COM.domain: 27201+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.631251 IP dns.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41249: 27201
NXDomain 0/1/0 (110)
02:30:15.631474 IP mailserver.Aus.Sun.COM.41250 > dns.Aus.Sun.COM.domain: 27202+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.631480 IP mailserver.Aus.Sun.COM.41250 > dns.Aus.Sun.COM.domain: 27202+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.632386 IP dns.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41250: 27202
NXDomain 0/1/0 (110)
02:30:15.633410 IP mailserver.Aus.Sun.COM.41251 > dns.Aus.Sun.COM.domain: 28805+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.633418 IP mailserver.Aus.Sun.COM.41251 > dns.Aus.Sun.COM.domain: 28805+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.634324 IP break.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41251: 28805
NXDomain 0/1/0 (110)
02:30:15.634526 IP mailserver.Aus.Sun.COM.41252 > dns.Aus.Sun.COM.domain: 28806+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.634531 IP mailserver.Aus.Sun.COM.41252 > dns.Aus.Sun.COM.domain: 28806+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:30:15.635325 IP break.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41252: 28806
NXDomain 0/1/0 (110)
```

(Using **$:S** or **$:A** modifier)

```
02:32:07.923587 IP mailserver.Aus.Sun.COM.41253 > dns.Aus.Sun.COM.domain: 63100+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:32:07.923599 IP mailserver.Aus.Sun.COM.41253 > dns.Aus.Sun.COM.domain: 63100+
TXT? 3.100.168.192.zen.spamhaus.org. (46)
02:32:07.924979 IP dns.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41253: 63100
NXDomain 0/1/0 (110)
02:32:07.927616 IP mailserver.Aus.Sun.COM.41254 > dns.Aus.Sun.COM.domain: 63101+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:32:07.927627 IP mailserver.Aus.Sun.COM.41254 > dns.Aus.Sun.COM.domain: 63101+
A? 3.100.168.192.zen.spamhaus.org. (46)
02:32:07.928609 IP dns.Aus.Sun.COM.domain > mailserver.Aus.Sun.COM.41254: 63101
NXDomain 0/1/0 (110)
```

- **Place rate-limiting mechanisms (metermaid, conn_throttle etc.) *before DNS RBL lookups***

If you use one of the email rate-limiting mechanisms e.g. MeterMaid or **conn_throttle.so**, placing these PORT_ACCESS rate-limiting lookups prior to the **dns_verify.so** lookup will help reduce the impact of a Denial of Service on Messaging Server. For more information, see the discussion on MeterMaid in *Messaging Server System Administrator's Guide*. e.g.

```
PORT_ACCESS

! TCP|server-address|server-port|client-address|client-port
  *|*|*|*|*  $C$|INTERNAL_IP;$3|$Y$E
  *|*|*|*|*  $C$:A$[IMTA_LIB:check_metermaid.so,throttle,ext_throttle,$3]$N421$
Connection$ declined$ at$ this$ time$E
  TCP|*|25|*|* $C$:S$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
   *  $YEXTERNAL
```

- **Use most successful lookup first (multiple lookups)**

By placing the RBL lookups in most-successful to least-successful order, the overall number DNS lookups will be reduced as Messaging Server will terminate the PORT_ACCESS mapping table processing after the first RBL lookup returns a DNS TXT or A record.

Adding the "$T" PORT_ACCESS mapping table flag to the **dns_verify.so** callout will provide additional logging information to help determine which RBL is the most successful e.g.

```
  TCP|*|25|*|* $C$:S$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
```

Adding "LOG_CONNECTION=7" to the Messaging Server MTA option.dat configuration file will result in an additional "T" record in the mail.log file when a connection is dropped due to the connecting host being listed in a DNS RBL e.g.

```
 12-Mar-2008 10:06:52.09 78f.4.686597 **           +      T
TCP|1.2.3.4|25|5.6.7.8|39802 571 http://www.spamhaus.org/query/bl?ip=5.6.7.8
```

In the above case the SpamHaus lookup returned a TXT record “http://www.spamhaus.org/query/bl?ip=5.6.7.8” which was returned to the connecting client.

By using this log information, and re-ordering the DNS RBL lookups to provide the best first-lookup match rate, your DNS lookups will be reduced therefore improving overall performance.

- **Don't use too many lookups**

If your site uses multiple DNS RBL lookups to increase the chances of blocking IP addresses that are known to send spam, reordering those rules as per the previous Hint/Tip may show that the latter lookups block negligible additional hosts and can therefore be removed.

- **Don't use DNS_VERIFY_DOMAIN dispatcher configuration**

The DNS_VERIFY_DOMAIN dispatcher option doesn't provide sufficient granularity and therefore **dns_verify.so** PORT_ACCESS lookups should be used instead as discussed throughout this guide.

- **Avoid lookups for known 'Friendly' IP ranges**

IP addresses for an organization can usually be split into three distinct categories:

=> Internal IP addresses of trusted email upload systems (e.g. other Messaging Server MTA relays, mailstores). These are usually defined in the INTERNAL_IP mapping table and which you never want to be blocked if the IP address of a host happens to be listed in Realtime Blacklist.

=> 'Friendly' IP addresses of trusted hosts which your organization has direct control over (e.g. user's PC), and can therefore take action to quarantine if the systems are found to be a source of spam email. These systems are unlikely to ever be listed on a DNS RBL blacklist and if they are you don't want them to be blocked. They are not trusted enough to consider 'Internal'.

=> External IP addresses which cannot be trusted and whose IP addresses definitely need to be verified against Realtime Blacklists.

To define a range of 'Friendly' IP addresses, add a new mapping table called FRIENDLY_IP, this table will have the same format as the INTERNAL_IP mapping table e.g.

```
FRIENDLY_IP

  $(192.168.100.0/24)  $Y
  *  $N
```

Add a new 'FRIENDLY_IP' check to the PORT_ACCESS mapping table. This check should be above any dns_verify.so lookups, but below any rate-limiting checks (to protect Messaging Server from Denial of Service attacks) e.g.

```
PORT_ACCESS

! TCP|server-address|server-port|client-address|client-port
  *|*|*|*|*  $C$|INTERNAL_IP;$3|$Y$E
  *|*|*|*|*  $C$:A$[IMTA_LIB:check_metermaid.so,throttle,ext_throttle,$3]$N421$
Connection$ declined$ at$ this$ time$E
  *|*|*|*|*  $C$|FRIENDLY_IP;$3|$YEXTERNAL$E
  TCP|*|25|*|* $C$:S$[IMTA_LIB:dns_verify.so,dns_verify_domain_
port,$1,zen.spamhaus.org.,Your$ host$ ($1)$ found$ on$ spamhaus.org$ RBLblock$
list]$T$E
  *  $YEXTERNAL
```

- **Have customers use 'submit' port or SSL port for sending emails**

The **dns_verify.so** lookups used in this guide are restricted to port 25 server connections only.

If a customer uploading emails (e.g. using Mozilla Thunderbird) to your Messaging Server uses a submit port (e.g. port 587) they will avoid the DNS RBL lookup – although they will still be required to authenticate so this mechanism does not provide a means of spammers to easily bypass the RBL checks.

Using the 'submit' port reduces the number of RBL checks that need to be performed and also stops your email customers from being accidentally blocked.

Improve Performance of DNS Lookups

If you need to perform a DNS RBL lookup, they should be as fast-as-possible to reduce the impact on overall email delivery and processing performance.

- **Use a local-caching name-server process**

A local-caching name-server will keep a local cache of DNS lookups; thus reducing network overhead (and delay) and reducing the impact of any network infrastructure/DNS infrastructure problems.

The following guides provide information on how to install and configure Bind 9 on the messaging server to operate as a caching name-server.

`http://www.logiqwest.com/dataCenter/Demos/RunBooks/DNS/DNSsetup.html`

- **Use local copy of Realtime Blacklist DNS tables**

Organizations such as SpamHaus provide the option to keep a local copy of the RBL DNS tables. This can then be used to provide a local copy of the RBL Blacklist data which is much faster and potentially more reliable then relying on an external DNS servers.

spamhaus.org data feed. (See `http://www.spamhaus.org/faq/`)

- **Use fast and reliable Realtime Blacklist DNS providers only**

Smaller DNS Realtime Blacklist providers may not have sufficient or local DNS mirrors to provide quick lookup times or they may be prone to periods of outages when heavily loaded.

Prior to using **any** RBL, make sure you search the Internet using your preferred web-search engine for any existing reviews, problems etc.

The consequences of an incorrect choice can be severe.

For example, the ordb.org RBL list shutdown in 2006. System administrators that didn't notice that the ordb.org list was no longer blocking emails received a rude shock on the 25th March 2008 when lookups using the ORDB list now returned a successful value for all lookups – therefore causing all emails to be blocked as a result.

Developing Messaging Server Architectural Strategies

Once you have identified your system performance needs, the next step in sizing your Messaging Server deployment is to size specific components based on your architectural decisions.

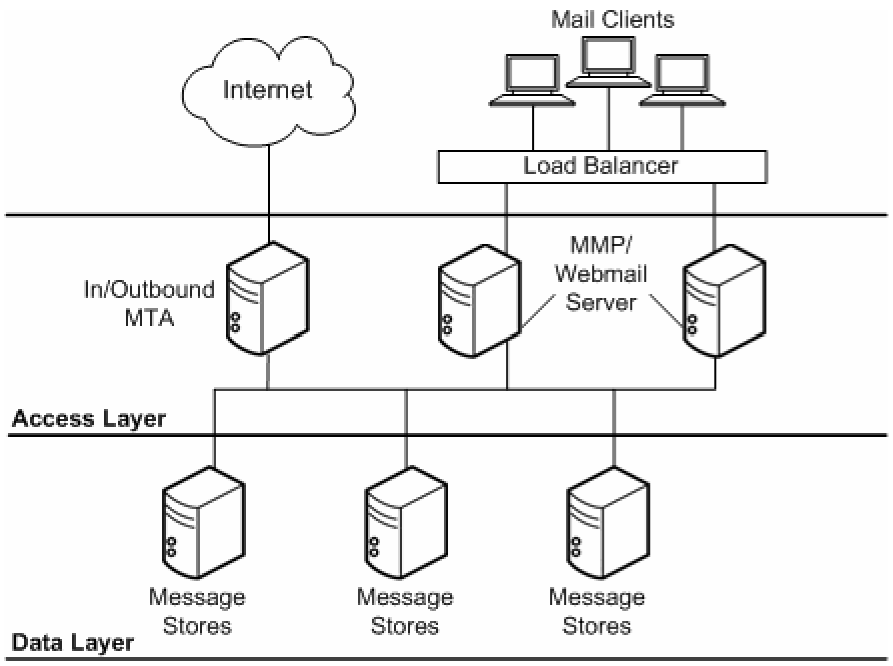
The following sections point out sizing considerations when you deploy two-tiered and one-tiered architectures.

Note: For more information, see the discussion on planning your architecture in "Developing a Messaging Server Architecture".

Two-tiered Messaging Server Architecture

A two-tiered architecture splits the Messaging Server deployment into two layers: an access layer and a data layer. In a simplified two-tiered deployment, you might add an MMP and an MTA to the access layer. The MMP acts as a proxy for POP and IMAP mail readers, and the MTA relays transmitted mail. The data layer holds the Message Store and Directory Server. Figure 4–1 shows a simplified two-tiered architecture.

Figure 4–1 Simplified Messaging Server Two-Tiered Architecture



Two-tiered architectures have advantages over one-tiered architectures that might impact your sizing decisions. Two-tiered architectures permit:

- Easier maintenance than one-tiered architectures
- Offloading of load-intensive processes like SSL, virus scanning, message reprocessing, and denial of service
- Easier growth management and system upgrade with limited overall downtime

The next several sections describe how to size specific components in a two-tiered deployment.

To Size the Message Store

The goals of sizing your Message Store are to identify the maximum number of concurrent connections your store can handle and to determine the number of messages that can be delivered to the store per second.

1. Determine the number of store machines and concurrent connections per machine based on the figures you gather by using a load simulator. For more information on sizing tools, see "Using a Messaging Server Load Simulator".

2. Determine the amount of storage needed for each store machine.
3. Use multiple store partitions or store machines, if it is appropriate for your backup and restoration of file system recovery times.

Consulting is often asked to specify a recommendation for the maximum number of users on a message store. Such a recommendation cannot be given without understanding:

- Usage patterns (as described in "Using a Messaging Server Load Simulator.")
- The maximum number of active users on any given piece of hardware within the deployment.
- Backup, restore, and recovery times. These times increase as the size of a message store increases.

To Size Inbound and Outbound MTAs

In general, separate your MTA services into inbound and outbound services. You can then size each in a similar fashion. The goal of sizing your MTAs is to determine the maximum number of messages that can be relayed per second.

To size inbound MTAs, you need to know the raw performance of your inbound MTA in a real-world environment.

1. From the raw performance of the inbound MTA, add SSL, virus scanning processes, and other extraordinary message processing.
2. Account for denial of service attacks at peak volume in the day.
3. Add enough MTAs for load balancing and for redundancy as appropriate. With redundancy, one or more of each type of machine can still handle peak load without a substantial impact to throughput or response time.
4. In addition, calculate sufficient disk capacity for network problems or non-functioning remote MTAs for transient messages.

To Size Your MMP

When you size your MMP, the calculation is based on your system load, particularly the number of POP and IMAP concurrent connections for the MMP.

In addition, you must:

1. Add more disks for an SMTP proxy.
2. Add capacity for load balancing and redundancy, if appropriate.

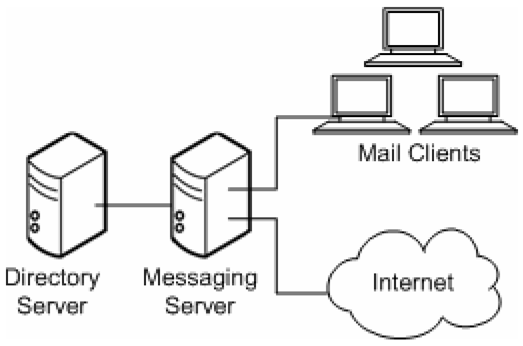
As with inbound MTA routers, one or more of each type of machine should still handle peak load without a substantial impact to throughput or response time when you plan for redundancy in your deployment.

Single-tiered Messaging Server Architecture

Note: This applies only to classic message store, and should only be used as a proof-of-concept or test deployment, not as an actual production deployment.

In a single-tiered architecture, there is no separation between access and data layers. The MTA, Message Store, and sometimes the Directory Server are installed in one layer. Figure 4–2 shows a single-tiered architecture.

Figure 4–2 Simplified Messaging Server Single-Tiered Architecture



Single-tiered architectures have lower up-front hardware costs than two-tiered architectures. However, if you choose a one-tiered architecture, you need to allow for significant maintenance windows.

To Size a Single-tiered Messaging Server Architecture

1. Size your message stores like you size message stores in a "Two-tiered Messaging Server Architecture".
2. Add CPU for SSL, if necessary.
3. Account for denial of service attacks.
4. Add more disks for the increased number of SMTP connections.
5. Add more disks for outbound MTA routing.

Note: For specific instructions on sizing Messaging components in single-tiered or two-tiered architectures, contact your Oracle representative.

Analyzing Your Messaging Server Requirements

Planning your Messaging Server deployment requires that you first analyze your organization’s business and technical requirements. This section helps you to gather and access your requirements, which you then use to determine you Messaging Server design.

Determining Messaging Server Project Goals

Your investigation and analysis should reveal your Messaging Server project's requirements. Next, you should be able to determine a clearly measurable set of goals. Specify these goals in such a manner that personnel not directly associated with the project can understand the goals and how to measure the project against them.

Stake holders need to accept the project goals. The project goals need to be measured in a post-implementation review to determine the success of the project.

Planning for Growth

In addition to determining what capacity you need today, assess what capacity you need in the future, within a time frame that you can plan for. Typically, a growth time line is in the range of 12 to 18 months (for a classic message store deployment). Growth expectations and changes in usage characteristics are factors that you need to take into account to accommodate growth.

As the number of users and messages increase, you should outline successful guidelines for capacity planning. You need to plan for increases in message traffic for the various servers, a larger volume of users, larger mailbox sizes, more calendar appointments, and so forth. As growth occurs in the user population, usage characteristics change over time. Your deployment goals (and therefore deployment design) must respond accordingly to be viable into the future.

Ideally, you should design your architecture to easily accommodate future growth. For example, use logical names for the Messaging Server services themselves. For more information, see the discussion on designing your deployment around logical service names in "Using Logical Service Names". Monitoring the deployment, once it enters its production phase, is also crucial to being able to understand when and by how much a deployment needs to grow.

Understanding Total Cost of Ownership

Total Cost of Ownership (TCO) is another factor that affects capacity planning. This includes choosing the hardware upon which to deploy Messaging Server. Table 4–4 presents some factors to consider as to whether to deploy more smaller hardware systems or fewer larger hardware systems.

Table 4–4 lists the considerations for total cost of ownership.

Table 4–4 Considerations for Total Cost of Ownership

Hardware Choices	Pros	Cons
More, smaller hardware systems	▪ Smaller hardware systems generally cost less. ▪ More, smaller hardware systems can be deployed across many locations to support a distributed business environment. ▪ More, smaller hardware systems can mean less down time for system maintenance, upgrade, and migration because traffic can be routed to other servers that are still online while others are being maintained.	▪ Smaller hardware systems have a more limited capacity, so more of them are needed. Management, administration, and maintenance costs go up as the number of hardware systems goes up. ▪ More, smaller hardware systems require more system maintenance because there are more of them to maintain.
Fewer, larger hardware systems	▪ Fewer hardware systems means fewer fixed management costs per server. If your management costs are a recurring monthly bill, whether internal or from an ISP, costs will be lower, because you have fewer hardware systems to manage. ▪ Fewer hardware systems can also mean easier system maintenance, upgrade, and migration because there are fewer systems to maintain.	▪ Larger hardware systems generally cost more initially. ▪ Fewer hardware systems can mean a greater system down-time for maintenance, upgrade and migration.

Identifying Messaging Server Deployment Goals

Before you purchase or deploy Messaging Server hardware or software, you need to identify your deployment goals. Deployment requirements can come from various sources within an organization. In many cases, requirements are expressed in vague terms, requiring you to clarify them towards determining a specific goal.

The outcome of your requirements analysis should be a clear, succinct, and measurable set of goals by which to gauge the deployment's success. Proceeding without clear goals that have been accepted by the stake holders of the project is precarious at best.

Some of the requirements you need to examine before you can plan your deployment include:

- Defining Business Requirements
- Defining Technical Requirements
- Defining Financial Requirements
- Defining Service Level Agreements (SLAs)

Defining Business Requirements

Your business objectives affect deployment decisions. Specifically, you need to understand your users' behavior, your site distribution, and the potential political issues that could affect your deployment. If you do not understand these business requirements, you can easily make wrong assumptions that impact the accuracy of your deployment design.

Operational Requirements

Express operational requirements as a set of functional requirements with straightforward goals. Typically, you might come across informal specifications for:

- End-user functionality
- End-user response times
- Availability/uptime
- Information archival and retention

For example, translate a requirement for adequate end-user response time into measurable terms such that all stake holders understand what is adequate and how the response time is measured.

Culture and Politics

A deployment needs to take into account your corporate culture and politics. Demands can arise from areas that end up representing a business requirement. For example:

- Some sites might require their own management of the deployed solution. Such demands can raise the project's training costs, complexities, and so forth.
- Given that the LDAP directory contains personnel data, the Human Resources department might want to own and control the directory.

Defining Technical Requirements

Technical requirements (or functional requirements) are the details of your organization's system needs.

Supporting Existing Usage Patterns

Express existing usage patterns as clearly measurable goals for the deployment to achieve. Here are some questions that will help you determine such goals.

- How are current services utilized?
- Can your users be categorized (for example, as sporadic, frequent, or heavy users)?
- How do users access services (from their desktop, from a shared PC or factory floor, from a roaming laptop)?
- What size messages do users commonly send?
- How many invitees are usually on calendar appointments?
- How many messages do users send?
- How many calendar events and tasks do users typically create per day or per hour?
- To which sites in your company do your users send messages?
- What level of concurrency, the number of users who can be connected at any given time, is necessary?

Study the users who will access your services. Factors such as when they will use existing services are keys to identifying your deployment requirements and therefore goals. If your organization's experience cannot provide these patterns, study the experience of other organizations to estimate your own.

Regions in organizations that have heavy usage might need their own servers. Generally, if your users are far away from the actual servers (with slow links), they will experience slower response times. Consider whether the response times will be acceptable.

Site Distribution

Use these questions to understand how site distribution impacts your deployment goals:

- How are your sites geographically distributed?
- What is the bandwidth between the sites?

Centralized approaches will require greater bandwidth than de-centralized. Mission critical sites might need their own servers.

Network Requirements

Here are some questions to help you understand your network requirements:

- Do you want to provide redundancy of network services?
- Do you want to limit available data on access layer hosts?
- Do you want to simplify end-user settings, for example, have end users enter a single mail host that does not have to change if you move them?
- Do you want to reduce network HTTP traffic?

Note: Answering yes to these questions suggests a two-tiered architecture.

Existing Infrastructure

You might be able to centralize servers if you have more reliable and higher available bandwidth.

- Will the existing infrastructure and facilities prove adequate to enable this deployment?
- Can the DNS server cope with the extra load? Directory Server? Network? Routers? Switches? Firewall?

Support Personnel

24-hour, seven-day-a-week (24 x 7) support might only be available at certain sites. A simpler architecture with fewer servers will be easier to support.

- Is there sufficient capacity in operations and technical support groups to facilitate this deployment?
- Can operations and technical support groups cope with the increased load during deployment phase?

Defining Financial Requirements

Financial restrictions impact how you construct your deployment. Financial requirements tend to be clearly defined from an overall perspective providing a limit or target of the deployment.

Beyond the obvious hardware, software, and maintenance costs, a number of other costs can impact the overall project cost, including:

- Training
- Upgrade of other services and facilities, for example, network bandwidth or routers
- Deployment costs, such as personnel and resources required to prove the deployment concept
- Operational costs, such as personnel to administer the deployed solution

You can avoid financial issues with the project by applying sufficient attention and analysis to the many factors associated with the project requirements.

Defining Service Level Agreements (SLAs)

You should develop SLAs for your deployment around such areas as uptime, response time, message delivery time, and disaster recovery. An SLA itself should account for such items as an overview of the system, the roles and responsibilities of support organizations, response times, how to measure service levels, change requests, and so forth.

Identifying your organization's expectations around system availability is key in determining the scope of your SLAs. System availability is often expressed as a percentage of the system uptime. A basic equation to calculate system availability is:

```
Availability = uptime / (uptime + downtime) * 100
```

For instance, a service level agreement uptime of four nines (99.99 percent) means that in a month the system can be unavailable for about four minutes.

Furthermore, system downtime is the total time the system is not available for use. This total includes not only unplanned downtime, such as hardware failures and network outages, but also planned downtime, preventive maintenance, software upgrade, patches, and so on. If the system is supposed to be available 7x24 (seven days a week, 24 hours a day), the architecture needs to include redundancy to avoid planned and unplanned downtime to ensure high availability.

5 Designing a Messaging Server Topology

This chapter provides information on how to design your messaging topology. A messaging topology describes the physical and logical layout of a networked messaging system. Specifically, a topology depicts the way the devices are arranged on a network and how they communicate with one another. In addition, a topology describes the way that data passes through a network. Topologies are bound to network protocols that direct the data flow.

For information about Messaging Server topology with the Cassandra message store, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Identifying Your Geographic Needs

The first step in designing your messaging topology is to identify your geographic needs. In particular, determine the messaging services you need to provide at each location within your organization:

1. Once you identify your deployment goals, determine the functions and features needed for each location within your deployment.
2. Understand your organization's physical constraints, specifically:
   - Available bandwidth
   - Distance between physical locations within your organization
   - Mail transaction rate and volume of mail storage at each physical location

Designing a Messaging Topology

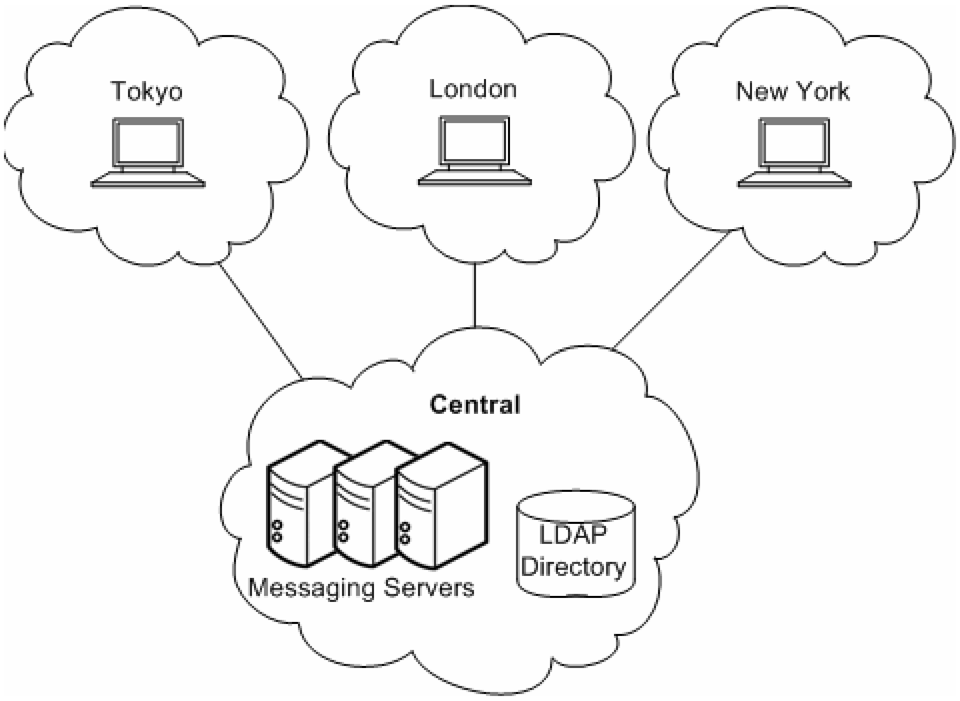
Before you develop your topology, you need a strategy to determine where you are going to put your messaging servers in your organization. Depending on your goals, there are four common topologies that you can apply to your organization:

- "Central Topology" consolidates most or all major system components and messaging servers at a single location.
- "Distributed Topology" spreads most or all system components and messaging servers across multiple sites.
- "Hybrid Topology" consolidates some system components and distributes other components across multiple locations.
- "Service Provider Topology" hosts multiple domains and handles larger customer base. Like a central topology, it consolidates most system components at a single location.

Central Topology

In a central topology, most or all major system components and messaging processes are located at one site. Clients at remote sites communicate over a Wide Area Network (WAN) to the centralized messaging servers. Figure 5–1 shows a central topology.

Figure 5–1 Central Topology



You should consider a central topology for your organization when:

- Messaging at remote sites is not mission critical.
- Users tend to send and receive small text messages.
- Your organization is located in one physical location or distributed across many small user populations.
- You do not have remote support personnel.
- Good bandwidth exists between remote sites and the central site (at least ISDN or better).

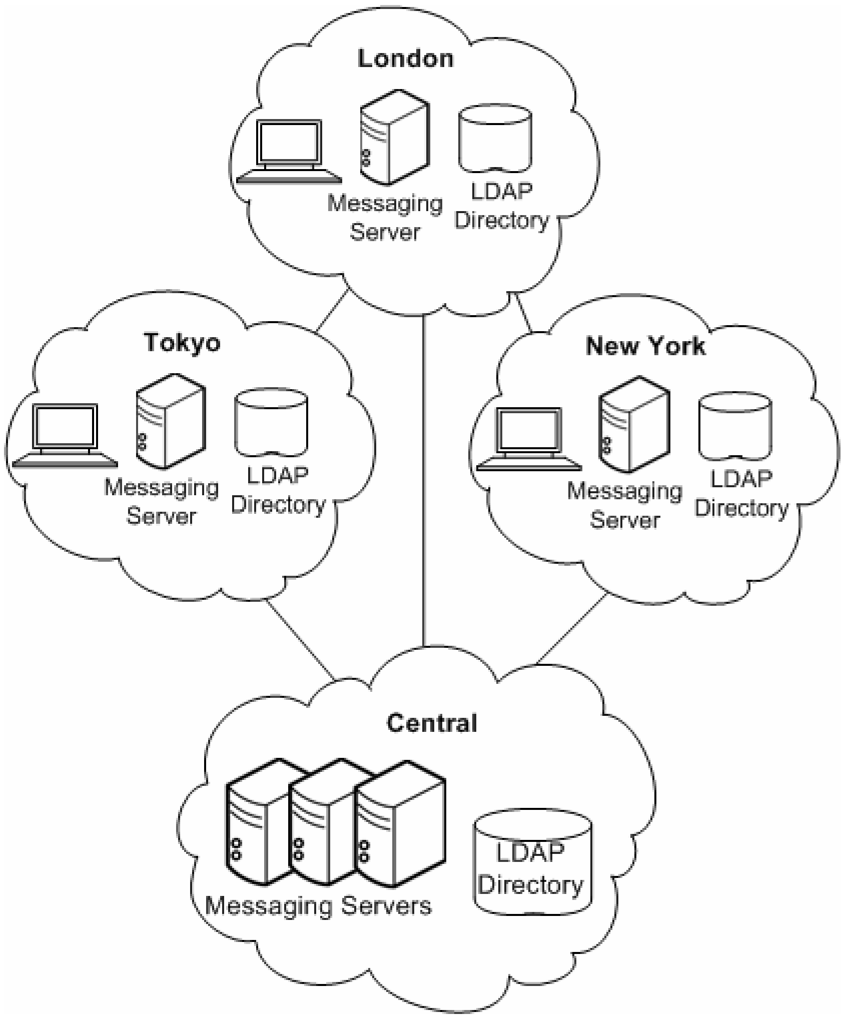
There are advantages to implementing a central topology. In general, a central topology has lower hardware and support costs. Central topologies tend to be easier to manage because you have a simplified messaging architecture and a directory replication structure with fewer replication agreements. With a simplified architecture and no need to coordinate installation among geographically distant sites, a central topology is faster to deploy.

That said, there are an equal number of disadvantages to implementing a central topology. A centralized approach heavily relies on a WAN. If the network does not function properly, users at the same site as well as users in remote locations could not send email to one another. Depending on network bandwidth and traffic, services might be slower during peak usage times. For users who send messages within the same domain, a central topology is inefficient. For example, looking at Figure 5–1, a message sent from one user in the Tokyo site would first travel to the Central site before being sent to another user in the Tokyo site.

Distributed Topology

In a distributed topology, most or all system components and messaging processes are distributed across multiple sites, usually at each remote site. Figure 5–2 shows a distributed topology.

Figure 5–2 Distributed Topology



You should consider a distributed topology for your site when:

- Messaging at remote sites is mission critical.
- Users send and receive large messages.
- You have large user populations at remote sites.
- Support personnel exists at remote sites.
- There is poor bandwidth to remote sites.

If bandwidth significantly impacts your topology strategy, you should consider upgrading the bandwidth. In general, bandwidth is relatively inexpensive. You might also consider a Virtual Private Networking (VPN), which uses existing high bandwidth Internet pipes rather than dedicated lines behind a firewall.

There are advantages to implementing a distributed topology. Users at regional sites have faster access to their messages because they do not have to retrieve messages over the WAN. Furthermore, messages sent within a regional location will incur less messaging traffic than in a central topology. However, satellite offices still rely on the WAN. Therefore, if lots of message traffic is generated in a satellite office, the WAN might need to be upgraded.

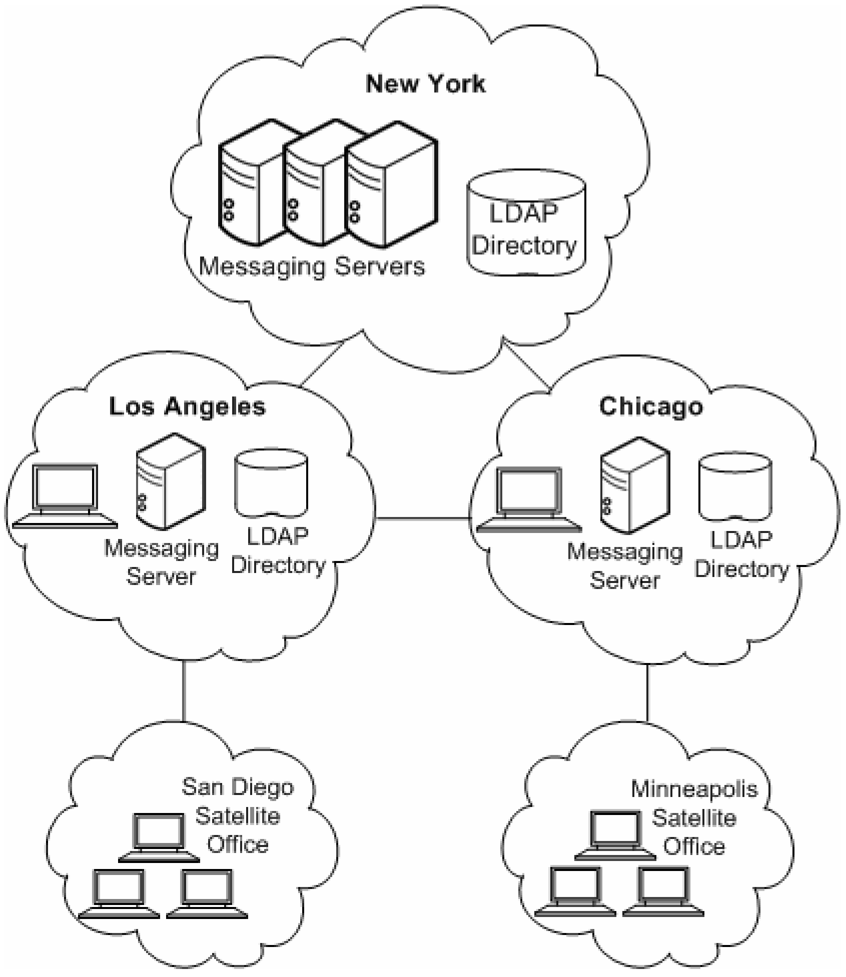
The disadvantages of implementing a distributed topology are that typically you will have higher hardware costs and higher support costs as you maintain more hardware at more locations. Support costs are also higher because of the complexity of the distributed topology. For example, failover in a distributed topology is more difficult to implement than in a central topology. In addition, it is much slower to initially deploy Oracle Communications Messaging Server because there are multiple servers spread across multiple sites.

Because Messaging Server accesses the LDAP directory, the LDAP server is a critical link in the mail delivery process. If you don't use remote LDAP replicas, and the central LDAP is down, the messaging service will not be usable.

Hybrid Topology

In a hybrid topology, central and distributed topologies are combined to meet the needs of an organization. Figure 5–3 shows a hybrid topology.

Figure 5–3 Hybrid Topology

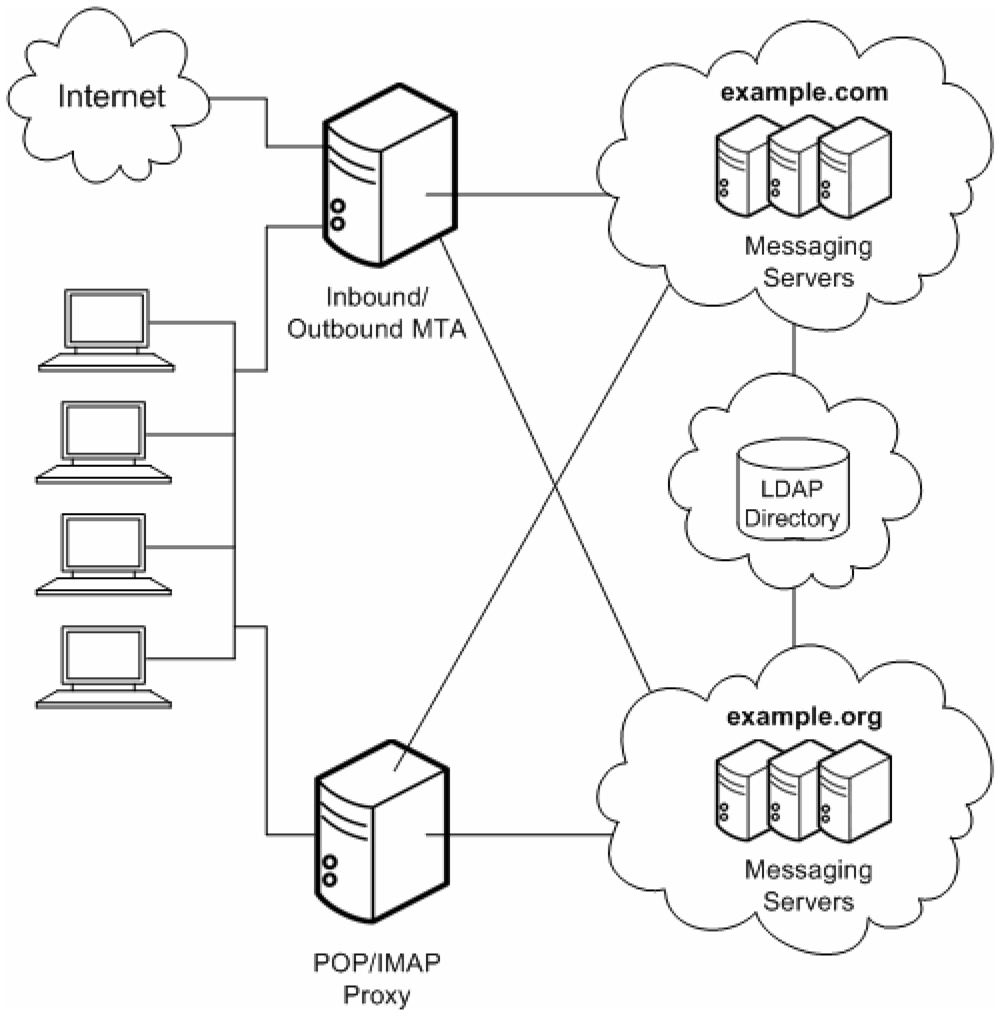


Organizations that benefit from a hybrid topology include those with many sites that have the ability to support a large user base. These sites that support them can house their own messaging servers. Some of these larger sites might have smaller satellite offices located in the general vicinity. But these satellite offices would not require their own messaging servers. Instead, the nearest major office would act as the central location for their services.

Service Provider Topology

In essence, a service provider topology is a large-scale central topology. Typically, a service provider hosts multiple domains and has a larger customer base than an enterprise. Systems are centralized and are able to support multiple users during peak hours. Figure 5–4 shows a service provider topology.

Figure 5–4 Server Provider Topology



Understanding Messaging Topology Elements

This section describes the most common elements in a messaging topology. Having some familiarity with the basic elements will make it easier for you to design your own topology.

The following topics are covered:

- Messaging Topology Components
- Using MTAs to Protect Your Messaging System
- Using MMPs
- Using Gateways

Messaging Topology Components

In "Designing a Messaging Topology," you were introduced to three components of a messaging topology: Messaging Server, Directory Server, and clients. This section will describe other components in a basic messaging topology.

- **Messaging Server.** Houses and maintains user mailboxes; it can also be a server that contains just the MTA portion of Messaging Server as described in **Internet-facing MTA** and **MTA Relay.**
- **Client.** Accesses messaging services from Messaging Server (often through the Messaging Multiplexor).
- **Directory Server.** Used by Messaging Server for name and alias lookup. Direct LDAP lookup determines where messages should be routed.
- **Messaging Multiplexor.** Connects clients to the appropriate Messaging Server for retrieving messages.
- **Internet-facing MTA.** Routes messages from the Internet and relays them across the firewall. Typically, a Messaging Server host is set up to perform this function.
- **MTA Relay.** The inbound MTA routes incoming messages to valid addresses in the appropriate Messaging Server. The outgoing MTA accepts outgoing messages from clients, queries LDAP to find out where to send the message, then sends it off to the appropriate server or out across the firewall to the Internet. Typically, a Messaging Server host is set up to perform this function.
- **DNS Server.** Resolves server names into IP addresses to allow messages to be routed to their proper address in the network.
- **Firewall.** Restricts Internet access of your internal site. You might even have a firewall between departments in your organization.

Using MTAs to Protect Your Messaging System

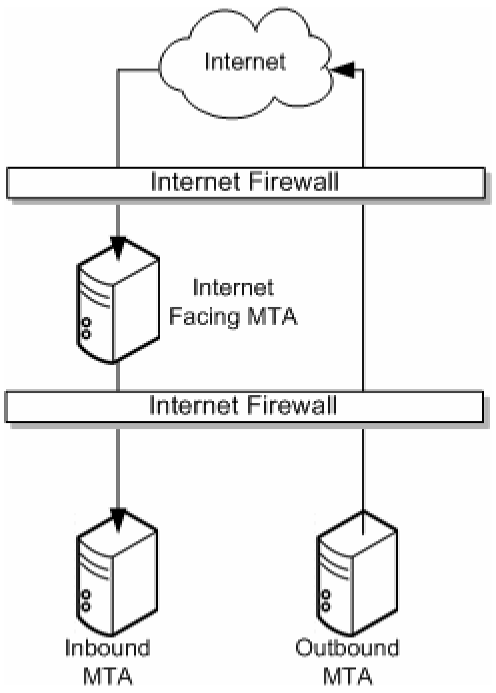
You can use MTAs to protect your Messaging Server deployment, as well as to control the flow of message traffic to and from your site.

An Internet-facing MTA is a single point of contact that receives messages from sites external to your organization. An Internet-facing MTA sends the incoming messages across the firewall to the inbound MTA, typically another Messaging Server.

The inbound MTA then queries the directory to determine where to send the message within the organization. The Internet-facing MTA is located in the demilitarized zone (DMZ) of the firewall (between the external and internal walls of the firewall), and does not have access to any information about servers other than the inbound MTA.

The outbound MTA accepts outgoing messages from clients. It queries LDAP to find out where to send the message, then sends it off to the appropriate server or out across the firewall to the Internet. This offloads the MTA work from messaging servers that are used by users to retrieve messages. Figure 5–5 illustrates the idea.

Figure 5–5 MTAs in Messaging Topology

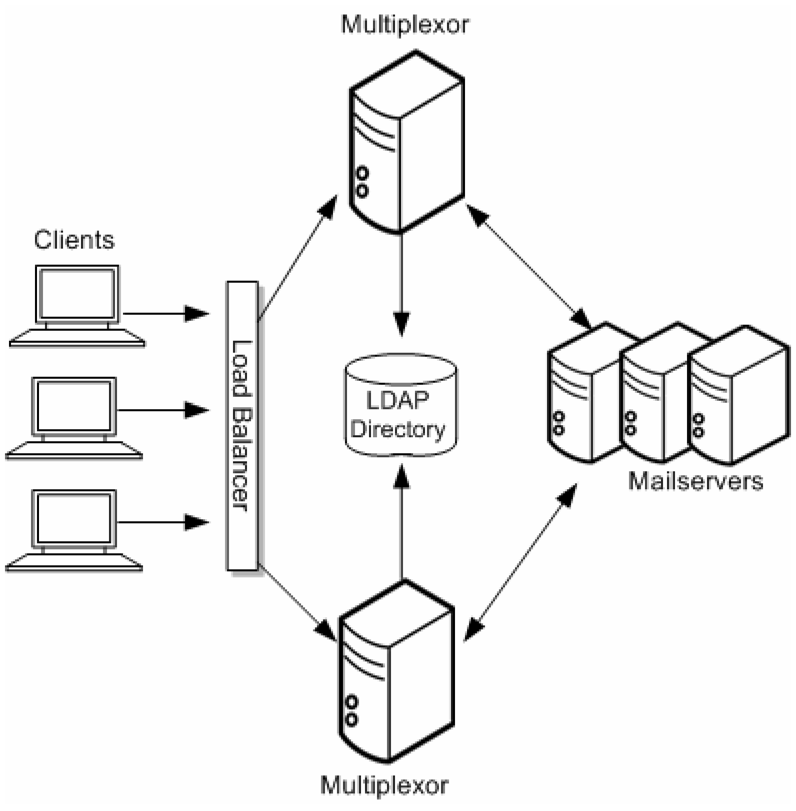


Using MMPs

The MMP enables you to mask the layout of your Messaging Server hosts from your end users. Consequently, you assign users to a generic MMP or a load balancer without having them point to the specific server where their mail boxes reside. Message access clients point to the MMP for retrieving incoming messages.

When such a client connects and authenticates, the MMP looks up the user information in the directory to determine where the user's messages are held. The MMP then connects the client to that specific server. Figure 5–6 shows how the MMP acts as a proxy for IMAP4 and POP3 connections to messaging servers.

Figure 5–6 MMP Overview



Use a load balancer in front of the multiple MMPs. It is unlikely that you would have a single MMP.

Using the MMP SMTP Proxy

Note: Starting with Messaging Server 8.0.2, the SMTP submission proxy in the MMP is no longer supported.

The MMP contains an SMTP proxy that is designed to accept messages but not transfer messages. Because of this design, never use the MMP SMTP Proxy as the target of a DNS MX record or to otherwise receive mail incoming from arbitrary sources on the Internet. Messaging Server does not support the use of the MMP SMTP Proxy in a message transfer capacity.

Messaging Server does support the use of the MMP SMTP proxy for message submission from end-user clients. However, the multiplexing functionality of the MMP, which is necessary to distribute POP and IMAP connections to the correct back-end store, is not necessary for SMTP submission. You can balance SMTP submission by MX records for mail clients that follow the standard, or by a simple load balancer for mail clients that do not follow the standard.

Only use the MMP SMTP Proxy in the following situations:

- If the MTA is becoming impeded with SSL/TLS processing, the MMP SMTP proxy can offload that processing for message submission while still supporting standard SMTP STARTTLS.
- If the MMP has SSL hardware acceleration for POP/IMAP, it might make sense to also leverage that for SMTP submission.

- If you need to use the "POP before SMTP" mechanism, then the MMP SMTP Proxy is required.
- The MMP SMTP proxy has a desired feature not present in the back-end MTA.
- If your deployment requires a proxy, then use the MMP SMTP proxy, which is specifically designed to preserve the security features and SMTP extensions present in the MTA and uses a custom SMTP extension **(XPEHLO)** to do so safely.

> **Note:** The MMP SMTP Proxy only works with Messaging Server's SMTP server as a back-end.

Using Gateways

Your organization might contain legacy messaging systems that use proprietary methods for messaging handling. Until you migrate your users, both messaging strategies must co-exist. To access these legacy systems, you can use an SMTP gateway, which enables SMTP connections between the new system and the other legacy systems. Usually legacy systems support SMTP connections so that the inbound MTA can route messages to it.

Creating a Messaging Topology Example

Once you have a basic understanding of your topological needs, your strategy, and the topology elements, you can create your messaging topology. To illustrate how to create a messaging topology, this section uses the example of the Example Corporation.

The Example Corporation is a multimedia organization headquartered in New York, with two smaller offices in Los Angeles and Chicago, and two satellite offices in San Diego and in Minneapolis.

Step 1: Identifying Messaging Goals

The first step in creating a topology is to understand the goals of your organization. Example's messaging goals can be categorized into business objectives, technical, and financial constraints.

Example's Business Objectives

The finance, marketing, legal, IT, and engineering groups are located in New York. The creative groups are located in Los Angeles and in San Diego. The technical support groups are located in Chicago and Minneapolis. Most messages are sent between Chicago, Los Angeles, and New York.

Employees at the Example Corporation rely on email as their primary method of communication. On average, employees send approximately 15 messages per day with attachments in the form of spreadsheets, presentations, or animation.

The deployment planners determined that Message Server hosts would be set up in Chicago, Los Angeles, and in New York. Since the volume of email traffic in San Diego and in Minneapolis is relatively light, these satellite offices will only have mail clients connecting to servers that are located in Chicago and in Los Angeles.

Example's Financial and Technical Constraints

Because of budgetary restrictions, the Example Corporation will be using the existing infrastructure and hardware that is already in place, moving servers to locations where there is critical need. 24x7 support will be available only in the New York, Chicago, and Los Angeles offices. All offices will be connected by T3 lines to the Internet.

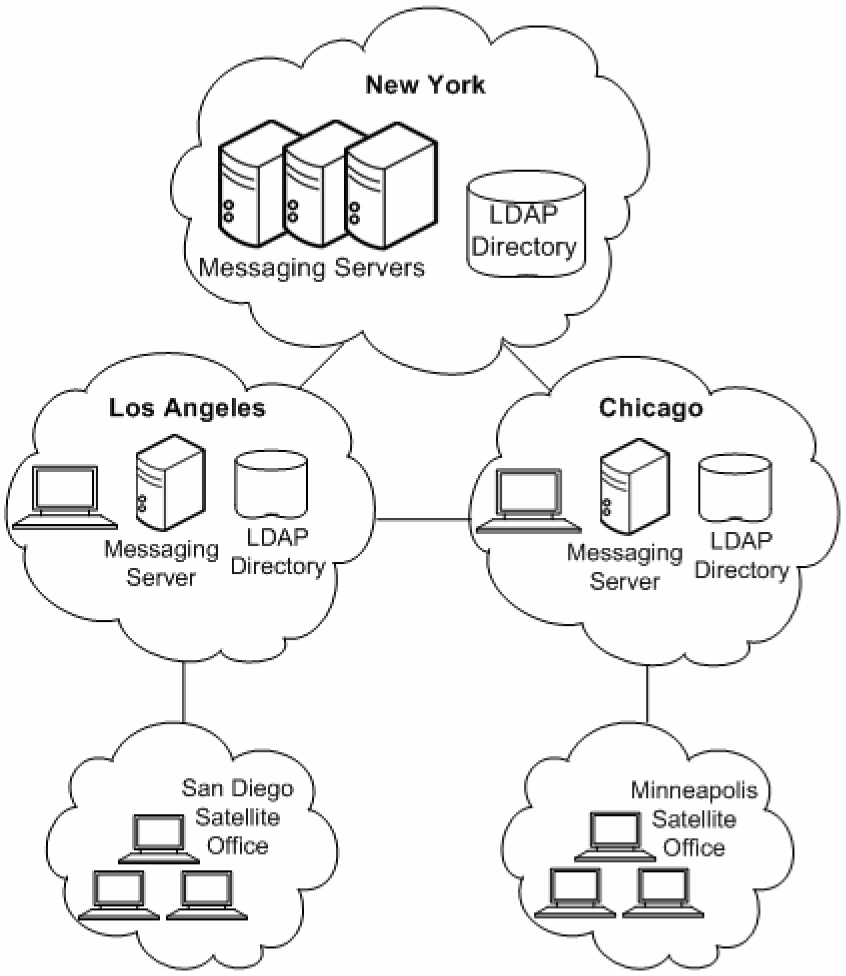
Step 2: Choosing a Topology Strategy

The second step in creating your messaging topology is to choose your topology strategy, described in "Designing a Messaging Topology". The Example Corporation evaluated their business objectives as well as their financial and technical constraints. They determined that:

- Messaging Server hosts did not need to be deployed at satellite sites, only mail clients.
- Good bandwidth exists at satellite sites (T3 lines).
- Regardless of location, mail users send and receive large messages throughout the corporation.
- There are large user populations in New York, Los Angeles, and Chicago, but not in Minneapolis or San Diego.
- Support personnel exist in New York, Los Angeles, and in Chicago.

The Example Corporation then mapped their objectives and constraints to a common design strategy. Figure 5–7 shows that the Example Corporation has chosen a hybrid topology.

Figure 5–7 Hybrid Topology for the Example Corporation

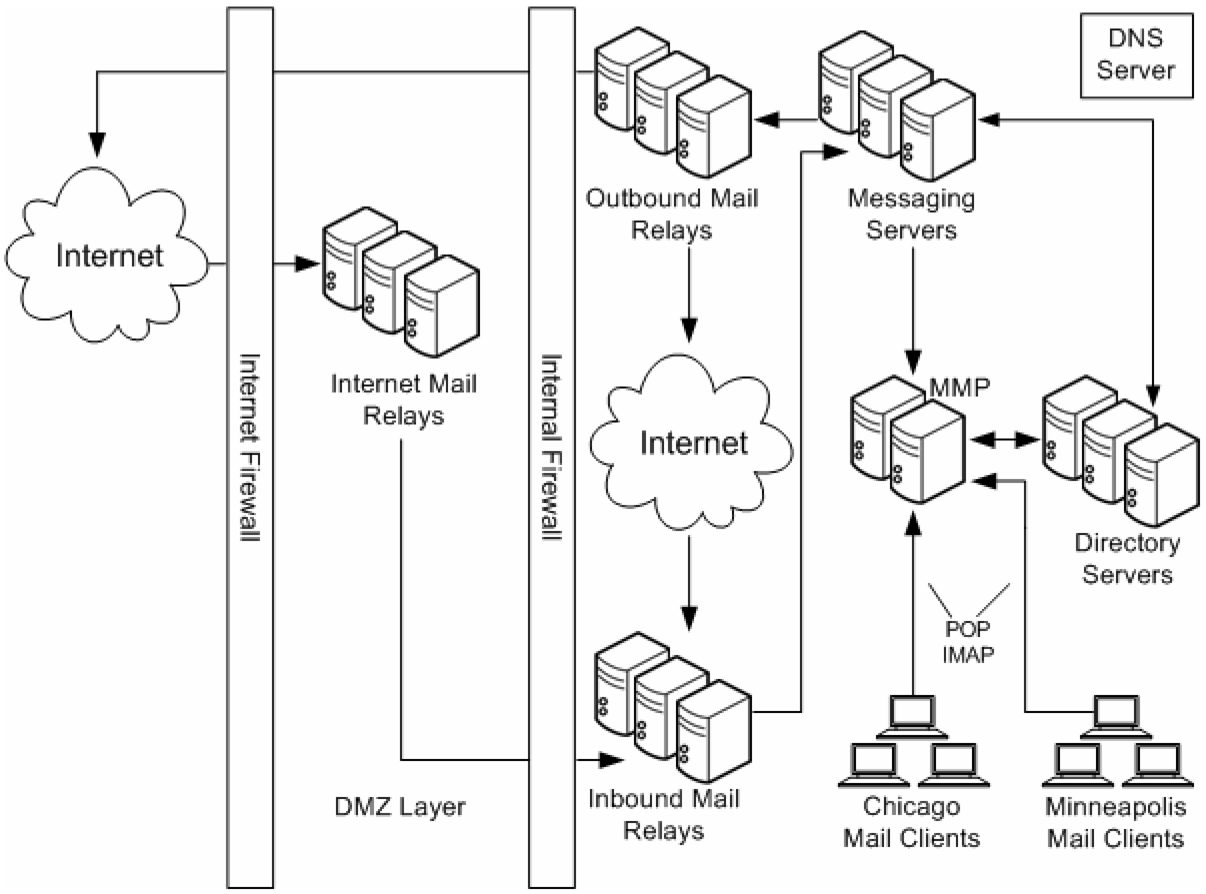


Because New York has the highest message transaction rate of messages entering and leaving the system, it has the most number of messaging servers. The smaller offices, Los Angeles and Chicago, also support San Diego and Minneapolis. However, these satellite offices do not require their own messaging servers. Instead, Chicago and Los Angeles act as the central location for their services.

Step 3: Planning the Topology Elements

The final step in creating your messaging topology is to plan your topology elements in your actual deployment, as described in "Understanding Messaging Topology Elements". Figure 5–8 illustrates the topology elements in the Chicago and Minneapolis offices.

Figure 5–8 Topological Elements in the Example Messaging Deployment for Chicago and Minneapolis



Because 30 percent of the workforce is made up of third-party vendors and contractors, internal firewalls are used in addition to the external firewalls in the topology to restrict access to locations within the company. Internet MTAs are placed in the topology to route messages from the Internet and relay them across the firewall. MTAs are added to route incoming and outgoing messages. Separating incoming and outgoing messages helps to manage the high volume of message traffic. The MMP connects employees' POP and IMAP mail clients to their mailboxes in the Messaging Servers. By using an MMP, employees do not have to know their specific mail host when they log in, and administrators can seamlessly move employees' mailboxes to different mail server locations.

Creating a messaging topology enables you to account for the physical and logical placement of all the elements in your deployment. Doing so ensures minimal rework of your installation.

Using Logical Service Names

Design your deployment around the use of logical names for Messaging Server servers. You should use logical names even on a single-system deployment, to position it for ease of future growth and expansion.Using logical names does not impose any additional deployment setup costs other than populating your DNS.

You can think of these logical names as falling into two categories: those that affect end users, such as setting in email client programs; and those affecting back-end administration, such as inbound SMTP servers.

Table 5–1, Table 5–2, and Table 5–3 describe these logical entities.

Table 5–1 User Facing Logical Names

Example	Description
mail.example.com	Name of the server from which end users collect their mail.
imap.example.com	Name of the IMAP server from which end users collect their mail.
pop.example.com	Name of the POP server from which the end users collect their mail.
smtp.example.com	Name of the SMTP server users set as outgoing mail server.

Table 5–2 Maintenance Level Logical Names

Example	Description
relay-in.example.com	Corresponds to a bank of inbound SMTP servers.
relay-out.example.com	Corresponds to a bank of outbound SMTP servers.
mmp.example.com	Corresponds to a bank of MMP servers.
storeAA.example.com	Back-end message store. Select a naming scheme to work with your topology, for example, **calstoreAA.example.com** through **calstoreZZ.example.com**.

Table 5–3 Mapping of User Level to Maintenance Level Logical Names

Maintenance Level	User Level
relay-in.example.com	Not applicable.
relay-out.example.com	**smtp.example.com**.
mmp.example.com	Any one or more of **mmp.example.com**, and **imap.example.com**.
storeAA.example.com - storeZZ.example.com	Not applicable, hidden from end users.
calstore_aa.example.com - calstore_az.example.com	Not applicable, hidden from end users.

6

Using a Configuration Management Tool with Messaging Server

This chapter describes how to use a configuration management tool with your Oracle Communications Messaging Server deployment.

About Using a Configuration Management Tool

Configuration management tools streamline the task of configuring and maintaining hosts in your deployment. Messaging Server does not supply its own tool for configuration management. However, with proper initial setup, you can use a third-party configuration management tool, such as Chef or Puppet, to maintain your Messaging Server deployment.

A configuration management tool makes sense to use when you have many Messaging Server hosts to install and maintain, and also when you deploy the Cassandra message store. For example, with Cassandra message store, reliability is provided by application-level redundancy across hosts. As a result, if a disk fails, you merely add a new Cassandra host, and remove the old one from the deployment. You would also use the same procedure to replace an entire host that is itself failing. A configuration management tool works well in these types of situations where you must add a new host to the deployment.

Creating the Environment to Use a Configuration Management Tool

When creating the proper Messaging Server environment in which to use a configuration management tool, you must first understand the role of the Messaging Server **configure** script. You use the **configure** script to create an initial, run-time configuration for a Messaging Server host. From that base working configuration, you then make specific customizations.

To summarize, the **configure** script:

1. Creates the Unix user and group for Messaging Server, if needed.
2. Creates Oracle Directory Server Enterprise Edition (Directory Server) entries (or LDIF) for the default domain, administrative user and postmaster, along with access control lists.
3. Creates Directory Server entries (or LDIF) for an administrative group and user for the host where **configure** is run; this administrative group and user have limited write access to the directory.
4. Creates *DataRoot* and *ConfigRoot* directories with correct permissions.

5. Creates subdirectories of the *DataRoot* and *ConfigRoot* directories with correct permissions.
6. Generates a random secret for authentication to local-only services, such as the job_controller and watcher.
7. Prompts you with the minimum number of questions about your environment to create an initial Messaging Server configuration with correct permissions.
8. Attempts to disable mail servers that come with the operating system, for example, Postfix and Sendmail.
9. Runs the **imsimta chbuild** command to create character set conversion tables.
10. Creates symbolic links (symlinks) from the **install** directory to the *DataRoot*, *ConfigRoot*, and **log** directories.

With Messaging Server 8.0.1 and prior, you must run the **configure** script on each Messaging Server host in your deployment. Starting with Messaging Server 8.0.2, the **configure** script has been modified so that you only need run **configure** once per deployment to perform the preceding steps 2 and 3. Also starting with Messaging Server 8.0.2, you do not need to perform the preceding step 9 (unless you customize character set tables), and the preceding steps 5 and 10 are optional.

Follow these best practices to use a configuration management tool with Messaging Server:

- The directory that contains the Messaging Server data files should always be **/var/***MessagingServer_home*. For example, when you install Messaging Server software in the default installation directory of **/opt/sun/comms/messaging64**, the data directory is **/var/opt/sun/comms/messaging64**. The Messaging Server configuration directory is created as a subdirectory of the **/var/opt/sun/comms/messaging64** directory, that is, **/var/opt/sun/comms/messaging64/config**.
- Use the configuration management tool to apply the appropriate operating system patches to your Messaging Server hosts.
- Use the configuration management tool to install Messaging Server software silently (that is, by using the **commpkg --silent** *INPUTFILE* command). Alternatively, install the Messaging Server software package directly by using either the **pkgadd** command (Solaris) or **rpm** command (Linux).
- Use the configuration management tool to perform steps 1 and 8 (for Messaging Server 8.0.1 and prior), or steps 1, 4, and 8 (Messaging Server 8.0.2) in the preceding section as appropriate.
- Use Unified Configuration (not legacy configuration) to configure and administer Messaging Server.
- For Messaging Server 8.0.1 and prior, use the configuration management tool to run the **configure** script. For Messaging Server 8.0.2, use the configuration management tool to perform step 4 in the preceding section, and to deploy the three configuration files (**config.xml**, **xpass.xml**, and **restricted.cnf**) that were created when you first ran the **configure** script for the deployment. Ensure that you correctly set the permissions on these files.
- As you change or update your initial Messaging Server configuration, use the configuration management tool to run **msconfig** recipes. Ensure that these recipes are idempotent (that is, running them a second time does not alter the configuration). It is helpful to use more than one **msconfig** recipe so that recipes shared by multiple roles (for example, configuring affinity groups) are separate

from recipes used by a single role (for example, enabling the MMP). If you use more than one recipe, ensure that each recipe modifies a separate part of the configuration to avoid ordering issues. For a good starting point, see the recipes provided in Messaging Server 8.0.2 for front-end and access-tier hosts (**CasFrontEnd.rcp** and **CasAccessTier.rcp**).

Considerations When Upgrading Messaging Server

In general, Messaging Server Unified Configuration is compatible with prior releases. If changes do occur to Unified Configuration in a new release, the idempotent **ConfigUpgrade.rcp** recipe performs these changes. The **ConfigUpgrade.rcp** recipe runs automatically the first time the **start-msg** command runs after a Messaging Server upgrade. When you evaluate a Messaging Server upgrade, you can manually run the **ConfigUpgrade.rcp** recipe on the configuration to see how the configuration has changed.

Considerations for Security

This section describes security issues to keep in mind when using a configuration management tool with Messaging Server.

Considerations for Directory Credentials

Use credentials that have limited write access to the directory (step 3 in "Creating the Environment to Use a Configuration Management Tool") to isolate directory security from Messaging Server security. However, using different LDAP credentials for each Messaging Server host does not significantly improve overall deployment security. As a result, there is no requirement to make changes to the Directory Server LDAP when adding a host to the Messaging Server deployment.

Considerations for Authentication to Local-only Services

Generate a different, random secret for local-only services for each Messaging Server host. If you choose not to run the **configure** script on each host in the deployment, then write an **msconfig** recipe, using the **strongrandom** recipe language function, to perform step 6 in "Creating the Environment to Use a Configuration Management Tool".

Considerations for Passwords

Use the configuration management tool's security capabilities to deploy passwords or credentials to your Messaging Server hosts. For example, Chef provides encrypted data bag items that you can use to protect files containing passwords or credentials. For Messaging Server, this includes statefiles (used for silent installation and configuration) and the **xpass.xml** file.

Considerations for ENS Authentication

Enable ENS authentication for the deployment. This is particularly important when affinity groups are used (as required by both Cassandra message store and classic store with automatic failover). For more information, see the topic about ENS support for password based authentication in *Messaging Server System Administrator's Guide*.

Considerations for PORT_ACCCESS Mapping

Use the PORT_ACCESS mapping table so that LMTP servers only accept connections from known LMTP clients. For more information about the PORT_ACCESS mapping table, see the topic on mapping tables in *Messaging Server System Administrator's Guide*.

Considerations for Firewalls

Use a deployment firewall to prevent external hosts from spoofing internal IP addresses and to block access to all ports except those intended to be externally visible. Setting up an external system that probes key internal ports (including LMTP, ENS, mshttpd, and WMAP) can be helpful to detect accidental firewall misconfiguration. More information about using a firewall with Messaging Server is available in *Messaging Server Security Guide*.

Considerations for SMTP Client IP Addresses

Having the IP address of SMTP clients available is important for the correct operation of anti-spam/anti-abuse mechanisms, and to track various forms of mail abuse. If you choose to terminate SMTP SSL/STARTTLS at a firewall or load balancer, that firewall or load balancer should support the SMTP XCLIENT extension (especially for enterprise deployments) and RFC 2979.

7 Messaging Server System Requirements

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

For information about Cassandra message store requirements, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Software Requirements

This section describes the software requirements for installing Messaging Server.

Supported Operating Systems

Table 7–1 lists operating systems that support Messaging Server. For all operating systems, we recommend that you run the latest software update with the latest recommended patch set.

Table 7–1 Supported Operating Systems

Operating System	CPU	Required Patches
Oracle Solaris 11.3.21 Oracle Solaris 10 1/13 (Update 11)	SPARC, x64	See the Oracle Solaris documentation for patch information.
Oracle Linux 7.2 Red Hat Linux 7.2 Oracle Linux 6.4 Red Hat Linux 6.4	x64	See the Oracle Linux and Red Hat Enterprise Linux documentation for patch information.

Supported High Availability Software

Table 7–2 lists high availability software support for Messaging Server.

Table 7–2 Supported High Availability Software

Product	Operating System	Version
Oracle Solaris Cluster	Solaris SPARC	3.0, 3.1 Update 4, 3.2, 3.3, 4.0, 4.1
Oracle Solaris Cluster	Solaris x86	3.1 Update 4, 3.2 U1, 3.3, 4.0, 4.1
Oracle Solaris Cluster	Red Hat Enterprise Linux	Not supported.
Oracle Solaris Cluster	Oracle Linux	Not supported.

Table 7–2 (Cont.) Supported High Availability Software

Product	Operating System	Version
Veritas	Solaris SPARC	3.5, 4.0, 4.1, 5.0, 6.0.1, 6.0.2
Veritas	Solaris x86	3.5, 4.0. 4.1, 5.0, 6.0.1, 6.0.2
Veritas	Red Hat Enterprise Linux	4.1, 5.0
Veritas	Oracle Linux	4.1, 5.5, 6.0, 6.0.1, 6.0.2
Oracle Clusterware	Solaris SPARC	12.1
Oracle Clusterware	Solaris x86	12.1
Oracle Clusterware	Red Hat Enterprise Linux	12.1
Oracle Clusterware	Oracle Linux	12.1

Required Software

Table 7–3 lists software required for installing and running Messaging Server.

Table 7–3 Software Requirements

Product	Version	Notes
Oracle Directory Server Enterprise Edition	Oracle Directory Server Enterprise Edition 6.x, 7, 11.x	For a fresh installation, use the latest version of Directory Server 11gR1.
Directory Server Setup Script (**comm_dssetup.pl**)	6.4.0.28 and higher	To prepare the LDAP directory for Messaging Server.

File System Recommendations

This section describes the file system requirements for installing Messaging Server.

Table 7–4 shows the file systems that are recommended for Messaging Server message stores.

Table 7–4 Messaging Server Clients

File System	Comments
LUFS (Logging UFS)	No comments.
VxFS (Veritas File System)	Veritas File System provides good system performance if configured properly. If you use VxVM, the Veritas Volume Manager, you need to watch carefully that the volumes and the log file for the volumes are set to be regularly striped.
HAStoragePlus File System for Oracle Solaris Cluster installations	The HAStoragePlus File System provides better performance than the default Oracle Solaris Cluster Global File System.
NFS (Network File System)	We support use of NFS as storage that is accessed by a single machine at a time. It can also be used to share autoreply and defragmentation histories between MTAs. See *Messaging Server System Administrator's Guide* for more information and setup details.
ZFS	See the topic on Messaging Server ZFS support in the *Messaging Server System Administrator's Guide* for more information.

Hardware Requirements

This section describes the hardware requirements for installing Messaging Server.

The number and configuration of the systems that you employ for your Messaging Server installation depends on the scale and the type 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 7–5 provides the minimum hardware requirements for Messaging Server.

Table 7–5 Minimum Hardware Requirements

Component	Requirement
Disk Space	Approximately 1 GB required for Messaging Server software. You also need adequate space for message store, database configuration directory, and log files, depending upon your site size.
RAM	1 GB

Time Synchronization Requirements

Configure Network Time Protocol (NTP) to ensure that time is synchronized across your deployment.

Information Requirements

This section describes the information needed before installing and configuring Messaging Server.

During the Messaging 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.

Messaging Server Information

Table 7–6 lists the Messaging Server information that you provide during initial configuration.

Table 7–6 Messaging Server Information

Information Type	Default Value	Comments
DataRoot	**/var/opt/sun/comms/messaging64**	No comments.
User name for server processes	**mailsrv**	No comments.
Group name for server processes	**mail**	If the user name for server processes already exists, then the primary group for that user name is used, and no option is prompted for.
Fully qualified host name of this system	FQDN of host	No comments.
Default mail domain name	Domain of host	No comments.

LDAP Information

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

Table 7–7 LDAP Information

Information Type	Default Value
Directory Server host name	Defaults to the loopback interface (where LDAP would live in a single-host evaluation deployment).
User/Group directory manager distinguished name (DN)	**cn=Directory Manager**
Directory manager password	No default value.

Postmaster, Administrator, and IP Relay Information

Table 7–8 lists the additional information that you provide during initial configuration.

Table 7–8 Notification Information

Information Type	Default Value
Mail address for postmaster notices	**admin@***domain*.
Password for server administration	No default.
Mail relay IP addresses (systems permitted to relay mail without authentication)	No default, but if nothing is provided, only connections from the host to itself can relay without authentication.

Directory Server Information

Table 7–9 lists the additional information that you provide during the Directory Server configuration with initial configuration.

Table 7–9 Directory Server Information

Option	Default Value
Instance Directory	**/var/opt/SUNWdsee/dsins1**
Directory Instance Port	389

Table 7–9 (Cont.) Directory Server Information

Option	Default Value
Directory instance SSL Port	636
Directory Manager DN	**cn=Directory Manager**
Directory Manager Password	Directory Manager password provided to Messaging Server initial configuration.

Front-End / Back-End Compatibility Matrix for Messaging Server Versions

This sections provides a compatibility matrix for front-end and back-end servers for different versions of Messaging Server for the various protocols. Table 7–10 lists the versions of the front-end, the protocol, and compatible back-end versions.

Table 7–10 Front-End / Back-End Compatibility Matrix for Messaging Server Versions

Front End	Protocol	Back End
MS 7/8 MMP *	IMAP/POP + SASL PLAIN	Standards Compliant IMAP/POP (including old MS versions)
MS 7/8 MTA	SMTP Relay	Standards Compliant SMTP servers
MS 7/8 MTA	Customized LMTP	MS 7/8 LMTP server
MS 7/8 mshttpd	SMTP Submit + SASL PLAIN	MS 7/8 MTA
MS 7.0.5.31.0/8 mshttpd **	IMAP + extensions	MS 8 IMAP server
MS 7/8 mshttpd	IMAP + extensions	MS 7 IMAP server
MS 8.0.1/8.0.2	Cassandra/classic store failover + IMAP, POP, LMTP	MS 8.0.2
MS 8.0.2 MMP/MTA/MSHTTP ***	Cassandra store round-robin + IMAP, POP, LMTP	MS 8.0.2 IMAP/POP/LMTP server

* Be sure to configure the MMP's IMAP capability string to be the subset of capabilities offered by all back-ends.

** Older versions of **mshttpd** have a bug that breaks compatibility with 8.0+ IMAP.

*** See **hostselect** proxy option in *Messaging Server Reference* for details.

8

Messaging Server Pre-Installation Tasks

This chapter provides information on the pre-installation tasks you must complete before you can install Oracle Communications Messaging Server.

For information about Cassandra message store pre-installation tasks, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Creating a UNIX System User and Group

System users run specific server processes, and privileges need to be given to these users so that they have appropriate permissions for the processes they are running.

Set up a system user account and group for Messaging Server, and set permissions for the directories and files owned by that user.

To Create a UNIX System User and Group

To create a UNIX system user and group, follow the steps below:

1. Log in as **root**.
2. Create a group name for server processes to which your system users belong.

   For example:

   ```
   groupadd mail
   ```

3. Create a user name for system processes and associate it with the group name you just created. In addition, set the password for that user.

   For example:

   ```
   useradd -g mail mailsrv
   ```

   The **useradd** and **usermod** commands are located in the **/usr/sbin** directory. See UNIX man pages for more information.

4. You might also need to check the **/etc/group** and **/etc/passwd** files to be sure that the user has been added to the system group that you created.

   Note: Should you decide not to set up UNIX system users and groups prior to installing Messaging Server, you are able to specify them when you run the configuration script. However, if the user name for server processes already exists, then the primary group for that user name is used, and the configuration script does not prompt for the option.

Checking the DNS Configuration

Check that DNS is running and configured properly for the Messaging Server host. The following example is for a host running Solaris 10 OS. The configuration is slightly different for a host running Solaris 11 OS.

To Check the DNS Configuration

To check the DNS configuration, follow the steps below:

1. Ensure that DNS is properly configured and that it is clearly specified how to route to hosts that are not on the local subnet.

   - The **/etc/defaultrouter** file should contain the IP address of the gateway system. This address must be on a local subnet.
   - The **/etc/resolv.conf** file exists and contains the proper entries for reachable DNS servers and domain suffixes.
   - In the **/etc/nsswitch.conf** file, the **hosts:** and **ipnodes:** line has the **files**, **dns** and **nis** keywords added. The keyword **files** must precede **dns** and **nis**. So if the lines look like this:

     ```
     hosts: nis dns files
     ipnodes: nis dns files
     ```

     They should be changed to this:

     ```
     hosts: files nis dns
     ipnodes: files nis dns
     ```

2. Make sure that the FQDN is the first host name specified after the IP address in the **/etc/hosts** file.

   If your Internet host table in your **/etc/hosts** file looks like this:

   ```
   123.456.78.910 budgie.west.example.com
   123.456.78.910 budgie loghost mailhost
   ```

   Change it so that there is only one line for the IP address of the host. Be sure the first host name is a fully qualified domain name. For example:

   ```
   123.456.78.910 budgie.west.example.com budgie loghost mailhost
   ```

   You can verify that the lines are read correctly by running the following commands:

   ```
   getent hosts ip_address
   getent ipnodes ip_address
   ```

   If the lines are read correctly, you should see the IP address followed by the FQDN and then the other values. For example:

   ```
   getent hosts 192.18.126.103
   192.18.126.103 budgie.west.example.com budgie loghost mailhost
   ```

Checking the Number of File Descriptors

The default installation of Linux uses 1024 file descriptors, which is insufficient for the correct operation of Messaging Sever. Messaging Server needs the maximum file descriptors set to 16384. Messaging Server processes rely heavily on multi-threading and this operating system restriction on the number of processes, if not changed as described below, can have random severe results.

To Check and Change the Number of File Descriptors

1. In a terminal window, as **root** user, verify the number of file descriptors.

   ```
   ulimit -n
   1024
   ```

   If this number is less than 16384, you need to increase the value.

2. To change the number of file descriptors, add the following to the /etc/sysctl.conf file. Or, if the parameter is already set, increase the value to 16384.

   ```
   fs.file-max = 16384
   ```

3. Add the following two lines to the **/etc/security/limits.conf** file.

   ```
   # Increase max file descriptors
   * - nofile 16384
   ```

   Note: Be careful when editing configuration files to not introduce extra spaces and lines, as this could cause errors. If you do accidentally type extra spaces, try copying existing lines and then overwriting them (by pasting) to get rid of any extra spaces. Using a text editor that displays hidden characters can also help.

4. Reboot the system for the value to take effect.

5. In a new terminal window, as **root** verify the change.

   ```
   ulimit -n
   16384
   ```

Installing Directory Server

Messaging 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 Messaging 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 Messaging Server Installer, adds the necessary Communications Suite schema to the LDAP. See "Preparing Directory Server" for more information.

Note: Always run the latest version of **comm_dssetup.pl** if you are upgrading any of the component products that depend on Directory Server.

Some LDAP object classes in the Communications Suite schema specifically support Messaging Server. For more information, see the discussion on understanding the schema that is used by Messaging Server in the *Schema Reference*.

9 Installing Messaging Server

This chapter describes how to install Oracle Communications Messaging Server. For more information on the Messaging Server install command, see the discussion on the **commpkg install** command in "commpkg Reference".

For information about installing Cassandra message store, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Before installing Messaging Server, read these chapters:

- Messaging Server Installation and Configuration Overview
- Planning Your Messaging Server Installation
- Messaging Server System Requirements
- Messaging Server Pre-Installation Tasks

About Messaging Server Components

When you install Messaging Server, you install and configure one or more of the following components:

- **Message Store.** Consists of a set of components that store, retrieve, and manipulate messages for mail clients.
- **Message Transfer Agent (MTA).** Receives, routes, transports, and delivers mail messages using the SMTP protocol. An MTA delivers messages to a local mailbox or to another MTA.
- **Messaging Multiplexor (MMP).** Enables scaling of the Message Store across multiple physical machines by decoupling the specific machine that contains a user's mailbox from its associated DNS name.
- **Webmail Server (mshttpd).** Acts as a front-end host that handles the HTTP protocol retrieval of messages from the message store. This component is used by Convergence to provide web-based access to end users.

Installation Assumptions

The instructions in this chapter assume:

- You are deploying Messaging Server on a single host or Solaris zone, or multiple hosts or Solaris zones.
- Each Messaging Server component is one functional component of your multi-host deployment.

- You are installing the Messaging Server component on a separate host or Solaris zone; you are not bundling the component with other Communications Suite products on the same host.
- Oracle Directory Server Enterprise Edition (Directory Server) is already installed.

The instructions also assume the following for the specific Messaging Server components:

- Message Store: If you are distributing multiple partitions of the message store across several hosts or zones, you can follow these instructions for each host on which you install store partitions.
- Message Transfer Agent (MTA): This MTA relay in and MTA relay out is one functional component of your multi-host deployment.
- Messaging Multiplexor (MMP): You are installing only the MMP front end; you are not installing message store or SMTP functions.
- Webmail Server (**mshttpd**): You are installing only the Webmail Server front end; you are not installing message store or SMTP functions.

About Unified Configuration

The recommendation is to use Unified Configuration for new Messaging Server installations. You must manage a Cassandra message store deployment by using Unified Configuration.

For more information, see the discussion on Unified Configuration in *Messaging Server System Administrator's Guide*.

Prerequisites for Installing Messaging Server

This section includes steps to take before installing Messaging Server. The topics in this section include:

- Before Installing Messaging Server
- Preparing Directory Server
- Configuring Messaging Server Against a Directory Server Replica

Before Installing Messaging Server

The following steps must be completed for all Messaging Server components: Message Store, Message Transfer Agent (MTA), Messaging Multiplexor (MMP), and Webmail Server (mshttpd).

1. Ensure that DNS is running and configured properly.

   For details, see "Checking the DNS Configuration".

2. Ensure you have sufficient file descriptors on Linux. For details, see "Checking the Number of File Descriptors".

3. Make sure you do not configure conflicting port numbers on a host when various components are running on a single machine.

   Table 9–1 lists the default port numbers used by Messaging Server.

Table 9–1 Messaging Server Default Ports

Port Number	Purpose
25	Standard SMTP port or MMP SMTP Proxy
110	Standard POP3 port or MMP POP3 Proxy
143	Standard IMAP4 port or MMP IMAP Proxy
225	Default port for communications to back-end store through LMTP
465	SMTP/SUBMIT over SSL or MMP SMTP Proxy over SSL
587	Standard Message Submission (SMTP SUBMIT) port
993	IMAP over SSL or MMP IMAP Proxy over SSL
995	POP3 over SSL or MMP POP Proxy over SSL
7997	Event Notification Service port
8990	**mshttpd** daemon port
8991	**mshttpd** over SSL daemon port
27442	Used by Job Controller for product internal communication
49994	Used by the Watcher for internal product communication

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".

Downloading the comm_dssetup.pl Script

1. Download the **comm_dssetup.pl** script from the Oracle software delivery website, located at:

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

   You can either download the Oracle Communications Directory Server Setup (**comm_dssetup.pl**) file separately, or as part of the Messaging Server software.

2. Copy the Directory Server Setup ZIP file to a temporary directory on your Directory Server hosts and extract the files.

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 root or become the superuser (root).
2. Start Directory Server, if necessary.
3. Copy the Comms DSsetup ZIP file to a temporary directory on your Directory Server hosts and extract the files.
4. Run the Installer.

   ```
   commpkg install
   ```

   For more information about running the Installer, see "commpkg Reference".
5. Select **Comms DSsetup** and proceed with the installation.

6. 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.

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

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

Configuring Messaging Server Against a Directory Server Replica

The following conditions might prevent you from configuring Messaging Server against a Directory Server host:

- You do not have Directory Server credentials.
- Messaging Server cannot communicate directly with the Directory Server master.

To configure your deployment to be able to run Messaging Server against a Directory Server replica, you must update the Directory Server master, which then feeds the replica with the necessary changes. You cannot update the Directory Server replica directly because the master Directory Server overwrites it.

To configure Messaging Server against a Directory Server replica:

1. Run the Messaging **configure** program using the replicated Directory Server credentials as described in "Configuring Messaging Server".

   Use the **--ldif** option to produce *MessagingServer_home*/**data/install/configure.ldif** file that is needed to allow proper privileges to the Directory Server.

2. Move the **configure.ldif** file to the Directory Server master.
3. Run the **ldapmodify** command on the **configure.ldif** file.

   Once the changes are replicated to the Directory Server replica, it is now configured to work with your Messaging Server.

Installing Messaging Server

The tasks to install Messaging Server are as follows:

- Downloading the Messaging Server Software
- Installing the Messaging Server Software

Downloading the Messaging Server Software

1. Download the media pack for Oracle Communications Messaging Server from the Oracle software delivery website, located at:

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

2. Copy the Messaging Server ZIP file to a temporary directory on your Messaging Server hosts and extract the files.

Installing the Messaging Server Software

For each Messaging Server component, you must install the Messaging Server software on each individual server host (Message Store, MTA, MMP, and Webmail Server).

To install the Messaging Server software:

1. On each server host (Message Server, MTA, MMP, and Webmail Server), log in as or become the superuser **(root)**.
2. Go to the directory where you extracted the Messaging Server ZIP file.
3. Run the installer.

   ```
   commpkg install
   ```

4. Choose the installation directory or accept the default.
5. Select **Messaging Server** and proceed with the installation.

When the installation is complete, continue with "Configuring Messaging Server".

Installing Messaging Server in Silent Mode

When you run the Messaging 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:

- To Run a Messaging Server Silent Installation
- About Upgrading Shared Components
- Silent Mode File Format

To Run a Messaging Server Silent Installation

1. Obtain the state file by one of the following two 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 the discussion on silent installation usage in "commpkg Reference".

> **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 9–2 shows which options you can change and provides examples:

Table 9–2 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 9–2 (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.	To specify 64-bit Messaging Server: COMPONENTS=MS64
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 Messaging Server on Solaris Zones

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

Installing on Solaris OS Zones: Best Practices

You can install Messaging Server components 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 Messaging 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."

- The one exception to these two guidelines is to install the HA agent (**MS_SCHA**) only in the global zone. The Messaging Server Installer automatically propagates HA agents to all non-global zones. That is, the **pkgadd -G** switch is not used for HA agents.

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 Messaging 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 Messaging Server on a single box with no zones. Simply install Messaging 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 Messaging Server in a non-global sparse root zone on Solaris 10. To install Messaging Server in a non-global sparse root zone, you first need to install/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 "Messaging 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/upgrade the OS patches and the Shared Components. Do not install Messaging Server components 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 Messaging 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 Messaging Server components in the sparse root zone.

Next Steps

After installing Messaging Server, continue with the following chapters:

- Follow the instructions in "Configuring Messaging Server" to finish installing and configuring Messaging Server and its individual components.
- Go to "Configuring Messaging Server for High Availability" if you need to configure Messaging Server for high availability.
- Follow the instructions in "Messaging Server Post-Installation Tasks" to perform post-installation tasks.

10 Configuring Messaging Server for High Availability

This chapter describes how to configure Oracle Communications Messaging Server for high availability.

For information about configuring Cassandra message store for high availability, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Designing for Service Availability

Once you have decided on your Messaging Server logical architecture, the next step is deciding what level of service availability is right for your site. The level of service availability you can expect is related to hardware chosen as well as the software infrastructure and maintenance practices you use. The following information discusses several choices, their value, and their costs.

High Availability Solutions Overview

Messaging Server supports three different high availability solutions, Oracle Solaris Cluster, Veritas Cluster Server (VCS), and Oracle Clusterware. Messaging Server provides agents for each of these solutions.

Messaging Server supports different cluster topologies. Refer to the appropriate cluster product documentation for more information. In addition, you can build in availability to your Messaging Server deployment by making infrastructure components, such as Directory Server, highly available.

The following topics in this section explain the options available for each component.

Automatic System Reconfiguration (ASR)

In addition to evaluating a purely highly available (HA) solution, you should consider deploying hardware that is capable of ASR.

ASR is a process by which hardware failure related downtime can be minimized. If a server is capable of ASR, it is possible that individual component failures in the hardware result in only minimal downtime. ASR enables the server to reboot itself and configure the failed components out of operation until they can be replaced. The downside is that a failed component that is taken out of service could result in a less performing system. For example, a CPU failure could result in a machine rebooting with fewer CPUs available. A system I/O board or chip failure could result in system with diminished or alternative I/O paths in use.

Different SPARC systems support very different levels of ASR. Some systems support no ASR, while others support very high levels. As a general rule, the more ASR capabilities a server has, the more it costs. In the absence of high availability software, choose machines with a significant amount of hardware redundancy and ASR capability for your data stores, assuming that it is not cost prohibitive.

Directory Server and High Availability

From a Messaging Server standpoint, the most important factor in planning your directory service is availability. As an infrastructure service, the directory must provide as near-continuous service as possible to the higher-level applications for authorization, access, email routing, and so forth.

A key feature of Directory Server that provides for high availability is replication. Replication is the mechanism that automatically copies directory data from one Directory Server to another. Replication enables you to provide a highly available directory service, and to geographically distribute your data. In practical terms, replication brings the following benefits:

- Failover
- Load balancing
- Higher performance and reduced response times
- Local data management

Table 10–1 shows how you can design your directory for availability.

Table 10–1 Designing Directory Server for High Availability

Method	Description
Single-master replication	A server acting as a supplier copies a master replica directly to one or more consumer servers. In this configuration, all directory modifications are made to the master replica stored on the supplier, and the consumers contain read-only copies of the data.
Two-way, multi-master replication	In a multi-master environment between two suppliers that share responsibility for the same data, you create two replication agreements. Supplier A and Supplier B each hold a master replica of the same data and there are two replication agreements governing the replication flow of this multi-master configuration.
Four-way multi-master	Provides a pair of Directory Server masters, usually in two separate data centers. This configuration uses four-way MultiMaster Replication (MMR) for replication. Thanks to its four-way master failover configuration, this fully-connected topology provides a highly-available solution that guarantees data integrity. When used with hubs in the replication topology, load distribution is facilitated, and the four consumers in each data center allow this topology to scale for read (lookup) operations.
Oracle Solaris Cluster Agent for Directory Server	Using Oracle Solaris Cluster software provides the highest level of availability for your directory implementation. In the case of failure of an active Directory Server node, Oracle Solaris Cluster provides for transparent failover of services to a backup node. However, the administrative (and hardware) costs of installing, configuring, and maintaining a cluster are typically higher than the Directory Server replication methods.

See the Directory Server Documentation for more information.

Messaging Server and High Availability

You can configure Messaging Server to be highly available by using clustering software. Messaging Server supports Oracle Solaris Cluster, Veritas Cluster Server, and Oracle Clusterware software.

In a tiered Messaging Server architecture, where front-end and back-end components are distributed onto separate machines, you would want to make the back-end components highly available through cluster technology as the back ends are the "stores" maintaining persistent data. Cluster technology is not typically warranted on the Messaging Server front ends as they do not hold persistent data. Typically, you would want to make the Messaging Server MTA and MMP, and Webmail Server front ends highly available through redundancy, that is, by deploying multiple front-end hosts. You could also add high availability to the MTA by protecting its disk subsystems through RAID technology.

For more information on Oracle Solaris Cluster topologies, see the discussion on key concepts for hardware service providers in *Oracle Solaris Cluster Concepts Guide for Solaris OS*.

Using Enabling Techniques and Technologies

In addition to the high availability solutions discussed in the previous sections, you can use enabling techniques and technologies to improve both availability and performance. These techniques and technologies include load balancers, Directory Proxy Server, and replica role promotion.

Using Load Balancers

You can use load balancers to ensure the functional availability of each tier in your architecture, providing high availability of the entire end-to-end system. Load balancers can be either a dedicated hardware appliance or a strictly software solution.

Load balancing is the best way to avoid a single application instance, server, or network as a single point of failure while at the same time improving the performance of the service. One of the primary goals of load balancing is to increase horizontal capacity of a service. For example, with a directory service, load balancers increase the aggregate number of simultaneous LDAP connections and LDAP operations per second that the directory service can handle.

Using Directory Proxy Server

Directory Proxy Server provides many proxy type features. One of these features is LDAP load balancing. Though Directory Proxy Server might not perform as well as dedicated load balancers, consider using it for failover, referral following, security, and mapping features.

See the Directory Proxy Server documentation for more information.

Using Replica Role Promotion

Directory Server includes a way of promoting and demoting the replica role of a directory instance. This feature enables you to promote a replica hub to a multi-master supplier or vice versa. You can also promote a consumer to the role of replica hub and vice versa. However, you cannot promote a consumer directly to a multi-master supplier or vice versa. In this case, the consumer must first become a replica hub and then it can be promoted from a hub to a multi-master replica. The same is true in the reverse direction.

Replica role promotion is useful in distributed deployments. Consider the case when you have six geographically dispersed sites. You would like to have a multi-master supplier at each site but are limited to only one per site for up to four sites. If you put at least one hub at each of the other two sites, you could promote them if one of the other multi-master suppliers is taken offline or decommissioned for some reason.

See the Directory Server documentation for more information.

Locating High Availability Product Reference Information

For more information on high availability models, see the following product documentation:

Oracle Solaris Cluster

- *Oracle Solaris Cluster Concepts Guide for Oracle Solaris OS*
- *Oracle Solaris Cluster Data Services Developer's Guide for Solaris OS*
- *Oracle Solaris Cluster Overview for Solaris OS*
- *Oracle Solaris Cluster System Administration Guide for Solaris OS*

Veritas Cluster Server

- *Veritas Cluster Server User's Guide*

Oracle Clusterware

- *Oracle Clusterware Administration and Deployment Guide*

Understanding Remote Site Failover

Remote site failover is the ability to bring up a service at a site that is WAN connected to the primary site in the event of a catastrophic failure to the primary site. There are several forms of remote site failover and they come at different costs.

For all cases of remote site failover, you need additional servers and storage capable of running all or part of the users' load for the service installed and configured at the remote site. By all or part, this means that some customers might have priority users and non-priority users. Such a situation exists for both ISPs and enterprises. ISPs might have premium subscribers, who pay more for this feature. Enterprises might have divisions that provide email to all of their employees but deem this level of support too expensive for some portion of those users. For example, an enterprise might choose to have remote site failover for mail for those users that are directly involved in customer support but not provide remote site failover for people who work the manufacturing line. Thus, the remote hardware must be capable of handling the load of the users that are allowed to access remote failover mail servers.

While restricting the usage to only a portion of the user base reduces the amount of redundant server and storage hardware needed, it also complicates configuration and management of fail back. Such a policy can also have other unexpected impacts on users in the long term. For instance, if a domain mail router is unavailable for 48 hours, the other MTA routers on the Internet will hold the mail destined for that domain. At some point, the mail will be delivered when the server comes back online. Further, if you do not configure all users in a failover remote site, then the MTA will be up and give permanent failures (bounces) for the users not configured. Lastly, if you configure mail for all users to be accepted, then you have to fail back all users or set up the MTA router to hold mail for the nonfunctional accounts while the failover is active and stream it back out once a failback has occurred.

Potential remote site failover solutions include:

- **Simple, less expensive scenario.** The remote site is not connected by large network bandwidth. Sufficient hardware is setup but not necessarily running. In fact, it might be used for some other purpose in the meantime. Backups from the primary site are shipped regularly to the remote site, but not necessarily restored. The expectation is that there will be some significant data loss and possibly a significant delay in getting old data back online. In the event of a failure at the primary site, the network change is manually started. Services are started, followed by beginning the **imsrestore** process. Lastly, the file system restore is started, after which services are brought up.
- **More complicated, more expensive solution.** Both Veritas and Oracle sell software solutions that cause all writes occurring on local (primary) volumes to also be written to remote sites. In normal production, the remote site is in lock step or near lock step with the primary site. Upon primary site failure, the secondary site can reset the network configurations and bring up services with very little to no data loss. In this scenario, there is no reason to do restores from tape. Any data that does not make the transition prior to the primary failure is lost, at least until failback or manual intervention occurs in the case of the MTA queued data. Veritas Site HA software is often used to detect the primary failure and reset the network and service bring up, but this is not required to get the higher level of data preservation. This solution requires a significant increase in the quantity of hardware at the primary site as there is a substantial impact in workload and latency on the servers to run the data copy.
- **Most available solution.** This solution is essentially the same as the software real time data copy solution except the data copy is not happening on the Message Store server. If the Message Store servers are connected to storage arrays supporting remote replication, then the data copy to the remote site can be handled by the storage array controller itself. Storage arrays that offer a remote replication feature tend to be large, so the base cost of obtaining this solution is higher than using lower-end storage products.

There are a variety of costs to these solutions, from hardware and software, to administrative, power, heat, and networking costs. These are all fairly straightforward to account for and calculate. Nevertheless, it is difficult to account for some costs: the cost of mistakes when putting a rarely practiced set of procedures in place, the inherent cost of downtime, the cost of data loss, and so forth. There are no fixed answers to these types of costs. For some customers, downtime and data loss are extremely expensive or totally unacceptable. For others, it is probably no more than an annoyance.

In doing remote site failover, you also need to ensure that the remote directory is at least as up to date as the messaging data you are planning to recover. If you are using a restore method for the remote site, the directory restore needs to be completed before beginning the message restore. Also, it is imperative that when users are removed from the system that they are only tagged as disabled in the directory. Do not remove users from the directory for at least as long as the messaging backup tapes that will be used might contain those users' data.

Questions for Remote Site Failover

Use the following questions to assist you in planning for remote site failover:

- What level of responsiveness does your site need?

  For some organizations, it is sufficient to use a scripted set of manual procedures in the event of a primary site failure. Others need the remote site to be active in rather short periods of time (minutes). For these organizations, the need for Veritas remote site failover software or some equivalent is overriding.

Note: Do not use both Oracle Solaris Cluster for local HA and Veritas software for remote site failover. Oracle Solaris Cluster does not support remote site failover at this time.

Also, do not allow the software to automatically failover from the primary site to the backup site. The possibility for false positive detection of failure of the primary site from the secondary site is too high. Instead, configure the software to monitor the primary site and alert you when it detects a failure. Then, confirm that the failure has happened before beginning the automated process of failing over to the backup site.

- How much data must be preserved and how quickly must it be made available?

  Although this seems like a simple question, the ramifications of the answer are large. Variations in scenarios, from the simple to the most complete, introduce quite a difference in terms of the costs for hardware, network data infrastructure, and maintenance.

Supported Versions of High-Availability Software in Messaging Server

For the latest supported versions and platforms, see "Supported Versions of High-Availability Software in Messaging Server."

Installation Methods for Messaging Server

A cluster agent is a Messaging Server program that runs under the cluster framework.

The Oracle Solaris Cluster Messaging Server agent is installed when you select Messaging Server Oracle Solaris Cluster HA Agent through the Messaging Server Installer.

1. Run the Messaging Server Installer command:

   ```
   commpkg install
   ```

   When prompted, select the Messaging Server Oracle Solaris Cluster HA Agent software.

2. Run the Oracle Solaris Cluster HA Agent pre-configuration command:

   ```
   cd MessagingServer_hahome/bin/
   init-config
   ```

Messaging Server Oracle Solaris Cluster HA Agent Initial Configuration

After installing the Messaging Server Oracle Solaris Cluster HA Agent software, you need to perform an initial configuration by running the following command:

```
MessagingServer_hahome/bin/init-config
```

This command registers the HA agent with the Oracle Solaris Cluster HA software. You must have the Oracle Solaris Cluster HA software installed prior to issuing this command.

Installing Messaging Server Oracle Solaris Cluster HA Agent in Solaris Zones

Oracle Solaris Cluster has added support for Oracle Solaris Zones. In this scenario, the Messaging Server Oracle Solaris Cluster HA agent should be installed in the global zone (and automatically installed in non-global zones). The Comms Installer will do this for you as long as you do the install in the global zone.

Take the following steps to install the Messaging Server Oracle Solaris Cluster HA agent in non-global zones:

1. Run the Messaging Server Installer command in the global zone only:

   ```
   commpkg install
   ```

   When prompted, select the Messaging Server Oracle Solaris Cluster HA Agent software. This command installs the Messaging Server Oracle Solaris Cluster HA Agent package on global zone and all non-global zones.

2. Run the Oracle Solaris Cluster HA Agent pre-configuration command in the global zone only:

   ```
   cd MessagingServer_hahome/bin/
   init-config
   ```

About High Availability Models

Messaging Server supports the following HA models:

- Asymmetric
- Symmetric
- N+1 (N Over 1)

Consult your HA documentation to determine which models your HA product supports.

Table 10–2 summarizes the advantages and disadvantages of each high availability model. Use this information to help you determine which model is right for your deployment.

Table 10–2 HA Model Summary and Recommendation

Model	Advantages	Disadvantages	Recommended Users
Asymmetric	Simple Configuration Backup node is 100 percent reserved	Machine resources are not fully utilized	A small service provider with plans to expand in the future
Symmetric	Better use of system resources Higher availability	Resource contention on the backup node HA requires fully redundant disks	A small corporate deployment that can accept performance penalties in the event of a single server failure
N + 1	Load distribution Easy expansion	Management and configuration complexity	A large service provider who requires distribution with no resource constraints

Table 10–3 illustrates the probability that on any given day the messaging service will be unavailable due to system failure. These calculations assume that on average, each server goes down for one day every three months due to either a system crash or

server hang, and that each storage device goes down one day every 12 months. These calculations also ignore the small probability of both nodes being down simultaneously.

Table 10–3 System Downtime Calculation

Model	Downtime Probability
Single server (no HA)	Pr(down) = (4 days of system down + 1 day of storage down)/365 = 1.37%
Asymmetric	Pr(down) = (0 days of system down + 1 day of storage down)/365 = 0.27%
Symmetric	Pr(down) = (0 days of system down + 0 days of storage down)/365 = (near 0)
N + 1	Pr(down) = (5 hours of system down + 1 day of storage down)/(365xN) = 0.27%/N

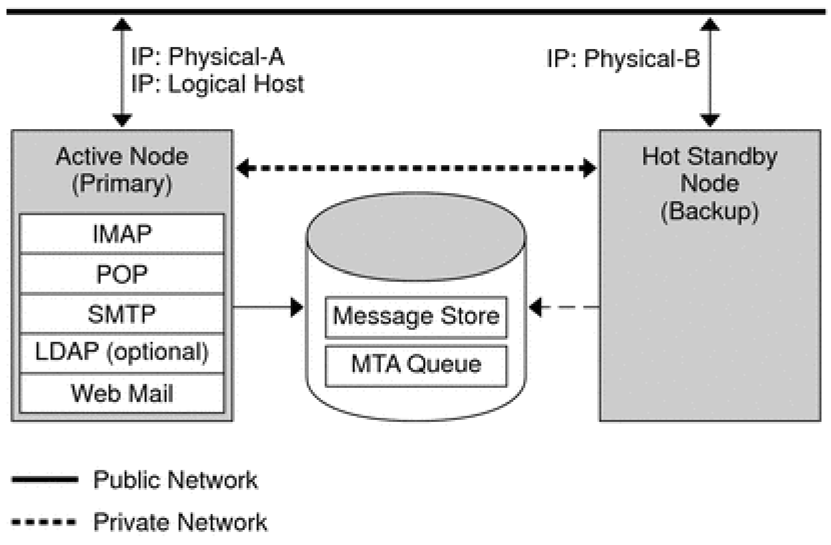
Asymmetric

The basic asymmetric or hot standby high availability model consists of two clustered host machines or nodes. A logical IP address and associated host name are designated to both nodes.

In this model, only one node is active at any given time; the backup or hot standby node remains idle most of the time. A single shared disk array between both nodes is configured and is mastered by the active or primary node. The message store partitions and MTA queues reside on this shared volume.

Figure 10–1 shows two physical nodes, Physical-A and Physical-B. Before failover, the active node is Physical-A. Upon failover, Physical-B becomes the active node and the shared volume is switched so that it is mastered by Physical-B. All services are stopped on Physical-A and started on Physical-B.

Figure 10–1 Asymmetric High Available Model



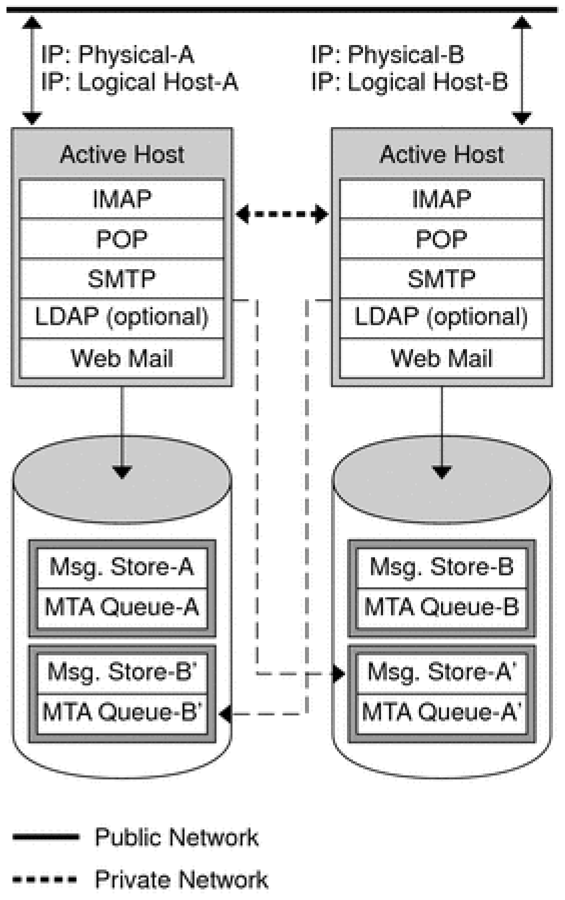
The advantage of this model is that the backup node is dedicated and completely reserved for the primary node. Additionally, there is no resource contention on the backup node when a failover occurs. However, this model also means that the backup node stays idle most of the time and this resource is therefore under utilized.

Symmetric

The basic symmetric or "dual services" high availability model consists of two hosting machines, each with its own logical IP address. Each logical node is associated with one physical node, and each physical node controls one disk array with two storage volumes. One volume is used for its local message store partitions and MTA queues, and the other is a mirror image of its partner's message store partitions and MTA queues.

Figure 10–2 shows the symmetric high availability mode. Both nodes are active concurrently, and each node serves as a backup node for the other. Under normal conditions, each node runs only one instance of Messaging Server.

Figure 10–2 Symmetric High Available Model



Upon failover, the services on the failing node are shut down and restarted on the backup node. At this point, the backup node is running Messaging Server for both nodes and is managing two separate volumes.

The advantage of this model is that both nodes are active simultaneously, thus fully utilizing machine resources. However, during a failure, the backup node will have more resource contention as it runs services for Messaging Server from both nodes. Therefore, you should repair the failed node as quickly as possible and switch the servers back to their dual services state.

This model also provides a backup storage array. In the event of a disk array failure, its redundant image can be picked up by the service on its backup node.

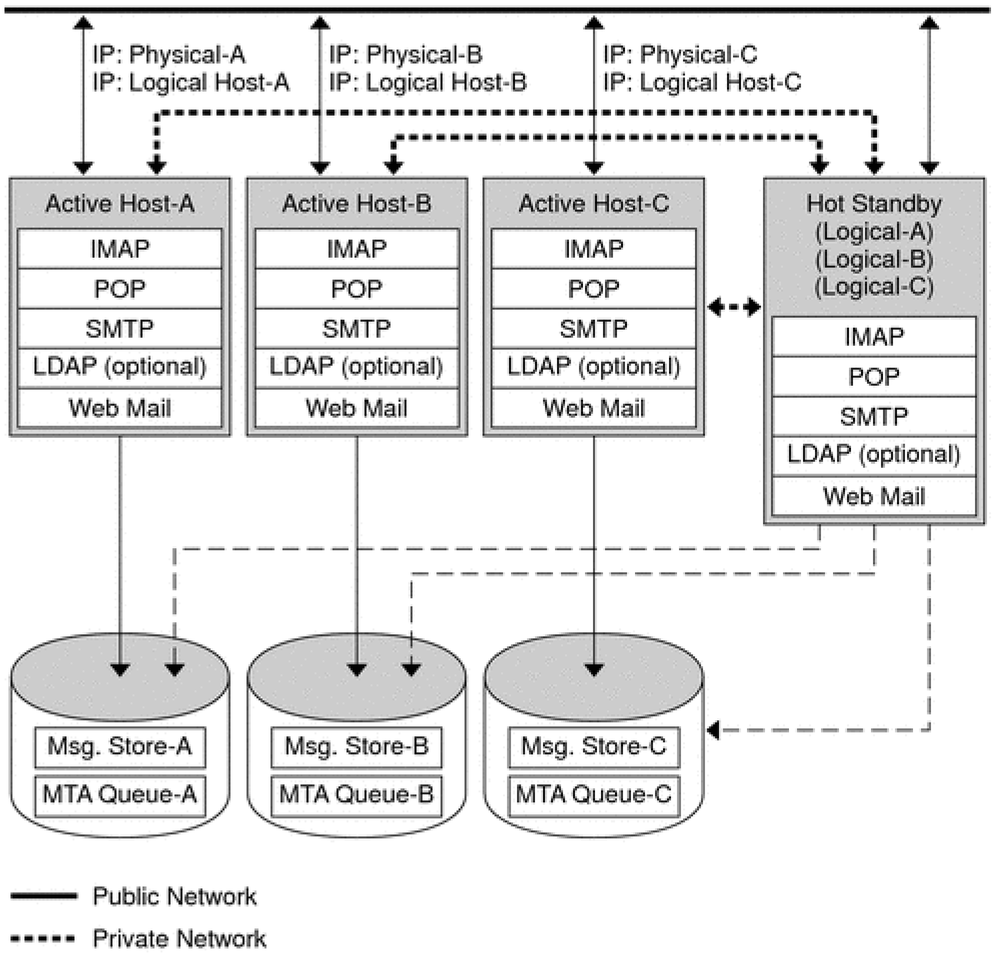
To configure a symmetric model, you need to install shared binaries on your shared disk. Note that doing so might prevent you from performing rolling upgrades, a feature that enables you to update your system during Messaging Server patch releases.

N+1 (N Over 1)

The N + 1 or "N over 1" model operates in a multi-node asymmetrical configuration. N logical host names and N shared disk arrays are required. A single backup node is reserved as a hot standby for all the other nodes. The backup node is capable of concurrently running Messaging Server from the N nodes.

Figure 10–3 illustrates the basic N + 1 high availability model.

Figure 10–3 N+1 High Available Model



Upon failover of one or more active nodes, the backup node picks up the failing node's responsibilities.

The advantages of the N + 1 model are that the server load can be distributed to multiple nodes and that only one backup node is necessary to sustain all the possible node failures. Thus, the machine idle ratio is 1/N as opposed to 1/1, as is the case in a single asymmetric model.

To configure an N+1 model, you need to install binaries only on the local disks (that is, not shared disks as with the symmetric model). The current Messaging Server installation and setup process forces you to put the binaries on the shared disk for any symmetric, 1+1, or N+1 asymmetrical or symmetrical HA solution.

Configuring Messaging Server Oracle Solaris Cluster HA Agent

To configure the Solaris Cluster HA agent:

1. On each node in the cluster create the Messaging Server runtime user and group under which the Messaging Server will run.

   The user ID and group ID numbers must be the same on all nodes in the cluster. The *runtime user ID* is the user name under which Messaging Server runs. This name should not be root. The default is **mailsrv**. The *runtime Group ID* is the group under which Messaging Server runs. The default is **mail**. Although the configure utility can create these names for you, you can also create them before running configure as part of the preparation of each node as described in this chapter. The runtime user and group ID names must be in the following files:

   - **mailsrv**, or the name you select, must in **/etc/passwd** on all nodes in the cluster
   - **mail**, or the name you select, must be in **/etc/group** on all nodes in the cluster

2. Add required resource types to Oracle Solaris Cluster.

   Configure Oracle Solaris Cluster to know about the resources types we will be using.

   To register Messaging Server as your resource use the following command:

   ```
   # scrgadm -a -t SUNW.ims
   ```

   To register HAStoragePlus as a resource type, use this command:

   ```
   # scrgadm -a -t SUNW.HAStoragePlus
   ```

   To do the same with HAStorage as a resource type, use this command:

   ```
   # scrgadm -a -t SUNW.HAStorage
   ```

3. Create a failover resource group for the Messaging Server.

   If you have not done so already, create a resource group and make it visible on the cluster nodes which will run the Messaging Server. The following command creates a resource group named **MAIL-RG**, making it visible on the cluster nodes mars and venus:

   ```
   # scrgadm -a -g MAIL-RG -h mars,venus
   ```

   You may, of course, use whatever name you wish for the resource group.

4. Create an HA logical host name resource and bring it on-line.

   If you have not done so, create and enable a resource for the HA logical host name placing that resource in the resource group. The following command does so using the logical host name **meadow**. Since the **-j** switch is omitted, the name of the resource created will also be **meadow**. **meadow** is the logical host name by which clients communicate with the services in the resource group.

   ```
   # scrgadm -a -L -g MAIL-RG -l meadow
   # scswitch -Z -g MAIL-RG
   ```

5. Create an HAStorage or HAStoragePlus resource.

Next, you need to create an HA Storage or HAStoragePlus resource type for the file systems on which Messaging Server is dependent. The following command creates an HAStoragePlus resource named **disk-rs**, and the file system **disk_sys_mount_point** is placed under its control:

```
# scrgadm -a -j disk-rs -g MAIL-RG \
-t SUNW.HAStoragePlus \
-x FilesystemMountPoints=disk_sys_mount_point-1, disk_sys_mount_point-2 -x
AffinityOn=True
```

SUNW.HAStoragePlus represents the device groups, cluster and local file systems which are to be used by one or more data service resources. One adds a resource of type **SUNW.HAStoragePlus** to a resource group and sets up dependencies between other resources and the **SUNW.HAStoragePlus** resource. These dependencies ensure that the data service resources are brought online after:

- All specified device services are available (and collocated if necessary)
- All specified file systems are mounted following their checks

The **FilesystemMountPoints** extension property allows for the specification of either global or local file systems. That is, file systems that are either accessible from all nodes of a cluster or from a single cluster node. Local file systems managed by a **SUNW.HAStoragePlus** resource are mounted on a single cluster node and require the underlying devices to be Oracle Solaris Cluster global devices. **SUNW.HAStoragePlus** resources specifying local file systems can only belong in a failover resource group with affinity switch overs enabled. These local file systems can therefore be termed failover file systems. Both local and global file system mount points can be specified together.

A file system whose mount point is present in the **FilesystemMountPoints** extension property is assumed to be local if its **/etc/vfstab** entry satisfies both of the following conditions:

- Non-global mount option
- Mount at boot flag is set to no

Note: Instances of the **SUNW.HAStoragePlus** resource type ignore the mount at boot flag for global file systems.

For the HAStoragePlus resource, the comma-separated list of **FilesystemMountPoints** are the mount points of the Cluster File Systems (CFS) or Failover File Systems (FFS) on which Messaging Server is dependent. In the above example, only two mount points, **disk_sys_mount_point-1** and **disk_sys_mount_point-2**, are specified. If one of the servers has additional file systems on which it is dependent, then you can create an additional HA storage resource and indicate this additional dependency in Step 15.

For HAStorage use the following:

```
# scrgadm -a -j disk-rs -g MAIL-RG \
-t SUNW.HAStorage
-x ServicePaths=disk_sys_mount_point-1, disk_sys_mount_point-2 -x
AffinityOn=True
```

For the HAStorage resource, the comma-separated list of **ServicePaths** are the mount points of the cluster file systems on which Messaging Server is dependent. In the above example, only two mount points, **disk_sys_mount_point-1** and **disk_**

sys_mount_point-2, are specified. If one of the servers has additional file systems on which it is dependent, then you can create an additional HA storage resource and indicate this additional dependency in Step 15.

6. Install the required Messaging Server packages on the primary node. Choose the **Configure Later** option.

   Use the Communications Suite installer to install the Messaging Server packages.

   For symmetric deployments: Install Messaging Server binaries and configuration data on files systems mounted on a shared disk of the Oracle Solaris Cluster. For example, for Messaging Server binaries could be under **/disk_sys_mount_point-1/SUNWmsgsr** and the configuration data could be under **/disk_sys_mount_point-2/config**.

   For asymmetric deployments: Install Messaging Server binaries on local file systems on each node of the Oracle Solaris Cluster. Install configuration data on a shared disk. For example, the configuration data could be under **/disk_sys_mount_point-2/config**.

7. Configure the Messaging Server. See "Running the Messaging Server Initial Configuration Script."

   In the initial runtime configuration, you are asked for the Fully Qualified Host Name. You must use the HA logical hostname instead of the physical hostname.

   Be sure to use the shared disk directory path of your HAStorage or HAStoragePlus resource.

8. Run the **ha_ip_config** script to set **service.listenaddr** and **service.http.smtphost** to configure the **dispatcher.cnf** and **job_controller.cnf** files for high availability.

   The script ensures that the logical IP address is set for these parameters and files, rather than the physical IP address. It also enables the watcher process (sets **local.watcher.enable** to **1**), and auto restart process (**local.autorestart** to **1**).

   The **ha_ip_config** script should only be run once on the primary node.

9. Fail the resource group from the primary to the secondary cluster node in order to ensure that the failover works properly.

   Manually fail the resource group over to another cluster node. (Be sure you have superuser privileges on the node to which you failover.)

   Use the **scstat** command to see what node the resource group is currently running on ("online" on). For instance, if it is online on mars, then fail it over to venus with the command:

   ```
   # scswitch -z -g MAIL-RG -h venus
   ```

   If you are upgrading your first node, use the Messaging Server Installer and then configure Messaging Server. You will then failover to the second node where you will install the Messaging Server package through the Communications Suite Installer, but you will not have to run the Initial Runtime Configuration Program again. Instead, you can use the **useconfig** utility.

10. Install the required Messaging Server packages on the secondary node. Choose the **Configure Later** option.

    After failing over to the second node, install the Messaging Server packages using the Communications Suite Installer.

    For symmetric deployments: Do not install Messaging Server.

For asymmetric deployments: Install Messaging Server binaries on local file systems on the local file system.

11. Run **useconfig** on the second node of the cluster.

    The **useconfig** utility allows you to share a single configuration between multiple nodes in an HA environment. You don't need to run the initial runtime configure program. Instead use the **useconfig** utility.

    See "Using the useconfig Utility"for more information

12. Create an HA Messaging Server resource.

    It is now time to create the HA Messaging Server resource and add it to the resource group. This resource is dependent upon the HA logical host name and HA disk resource.

    In creating the HA Messaging Server resource, we need to indicate the path to the Messaging Server top-level directory: the **msg-svr-base** path. These are done with the **IMS_serverroot** extension properties as shown in the following command.

    ```
    # scrgadm -a -j mail-rs -t SUNW.ims -g MAIL-RG \
          -x IMS_serverroot=msg-svr-base \
          -y Resource_dependencies=disk-rs,meadow
    ```

    The preceding command creates an HA Messaging Server resource named **mail-rs** for the Messaging Server, which is installed on **IMS_serverroot** in the **msg-svr-base** directory. The HA Messaging Server resource is dependent upon the HA disk resource **disk-rs** as well as the HA logical host name meadow.

    If the Messaging Server has additional file system dependencies, then you can create an additional HA storage resource for those file systems. Be sure to include that additional HA storage resource name in the **Resource_dependencies** option of the above command.

13. Enable the Messaging Server resource.

    It is now time to activate the HA Messaging Server resource, thereby bringing the messaging server online. To do this, use the command

    ```
    # scswitch -e -j mail-rs
    ```

    The above command enables the mail-rs resource of the MAIL-RG resource group. Since the MAIL-RG resource was previously brought online, the above command also brings **mail-rs** online.

14. Verify that things are working.

    Use the **scstat -pvv** command to see if the MAIL-RG resource group is online.

Unconfiguring Messaging Server HA Support

This section describes the high-level steps to unconfigure a simple HA configuration for Oracle Solaris Cluster. The exact procedure may differ for your deployment, but follows the same logical order described below.

1. Become the superuser.

   All of the following Oracle Solaris Cluster commands require that you be running as user superuser.

2. Bring the resource group offline.

   To shut down all of the resources in the resource group, issue the command

```
# scswitch -F -g MAIL-RG
```

This shuts down all resources within the resource group (for example, the Messaging Server and the HA logical host name).

3. Disable the individual resources.

   Next, remove the resources one-by-one from the resource group with the commands:

   ```
   # scswitch -n -j mail-rs
   # scswitch -n -j disk-rs
   # scswitch -n -j budgie
   ```

4. Remove the individual resources from the resource group.

   Once the resources have been disabled, you may remove them one-by-one from the resource group with the commands:

   ```
   # scrgadm -r -j mail-rs
   # scrgadm -r -j disk-rs
   # scrgadm -r -j budgie
   ```

5. Remove the resource group.

   Once the all the resources have been removed from the resource group, the resource group itself may be removed with the command:

   ```
   # scrgadm -r -g MAIL-RG
   ```

6. (Optional) Remove the resource types.

   Should you need to remove the resource types from the cluster, issue the commands:

   ```
   # scrgadm -r -t SUNW.ims
   # scrgadm -r -t SUNW.HAStoragePlus
   ```

Veritas Cluster Server Agent Installation

Messaging Server can be configured with Veritas Cluster Server 3.5, 4.0, 4.1, 5.0, and 6.0.2. Be sure to review the Veritas Cluster Server documentation prior to following these procedures. Veritas cluster Server agent for Messaging Server is part of the Messaging Server core package and is installed during Messaging Server installation only.

Veritas Cluster Server Requirements

Veritas Cluster software is already installed and configured as described in the following instructions along with the Messaging Server software on both nodes.

VCS Installation and Configuration Notes

The following instructions describe how to configure Messaging Server as an HA service, by using Veritas Cluster Server. The default **main.cf** configuration file sets up a resource group called **ClusterService** that launches the **VCSweb** application. This group includes network logical host IP resources like **csgnic** and **webip**. In addition, the **ntfr** resource is created for event notification.

To Configure Messaging Server as an HA Service by Using Veritas Cluster Server

These Veritas Cluster Server instructions assume you are using the graphical user interface to configure Messaging Server as an HA service.

1. Launch Cluster Explorer from one of the nodes.

   To launch Cluster Explorer, run the following command:

   ```
   /opt/VRTSvcs/bin/hagui
   ```

   The **VRTScscm** package must be installed to use the GUI.

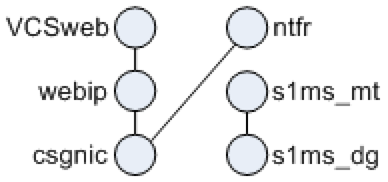
2. Using the Cluster Explorer, add a service group called **MAIL-RG**.
3. Add **s1ms_dg** disk group resource of type **DiskGroup** to the service group **MAIL-RG** and enable it.
4. Add **s1ms_mt** mount resource of type **Mount** to the service group **MAIL-RG**.

   Click the Link button to enable linking resources, if they are not already enabled.

5. Create a link between **s1ms_mt** and **s1ms_dg**.
6. Enable the resource **s1ms_mt**.

   Figure 10–4 depicts the dependency tree:

Figure 10–4 Veritas Cluster Dependencies



7. Run the Messaging Server Installer to install the Messaging Server software.
   a. Run the Messaging Server Initial Runtime Configuration (**configure**) from the primary node (for example, **Node_A**) to configure Messaging Server. The initial runtime configuration program asks for the Fully Qualified Host Name. Enter the logical hostname. The program also asks to specify a configuration directory. Enter mount point of the file system related to shared disk.
   b. Messaging Server running on a server requires that the correct IP address binds it. This is required for proper configuration of Messaging in an HA environment. Execute **ha_ip_config** command to bind to correct IP address.

      ```
      MessagingServer_home/bin/ha_ip_config
      ```

      The **ha_ip_config** program asks for the Logical IP address and Messaging Server Base (*MessagingServer_home*).
   c. During Messaging Server installation, VCS agent related directory **vcsha** is created under the Messaging Server base directory, which will have VCS HA agent related files. Run **config-vcsha** to copy agent files to VCS configuration.

      ```
      MessagingServer_home/bin/config-vcsha
      ```

      Messaging Server and the Veritas agent are available on **Node_A**.
8. Switch to the backup node (for example, **Node_B**).
9. Run the Messaging Server Installer to install Messaging Server software on the backup node (**Node_B**).

10. After installing Messaging Server, use the **useconfig** utility to obviate the need for creating an additional initial runtime configuration on the backup node (**Node_B**).

    The **useconfig** utility enables you to share a single configuration between multiple nodes in an HA environment. This utility is not meant to upgrade or update an existing configuration. To enable the utility, run **useconfig** to point to your previous Messaging Server configuration:

    ```
    MessagingServer_home/bin/useconfig
    MessagingServer_home/config
    ```

11. As VCS HA agent is part of Messaging Server installation, run **config-vcsha** to copy agent files to VCS configuration.

    ```
    MessagingServer_home/bin/config-vcsha
    ```

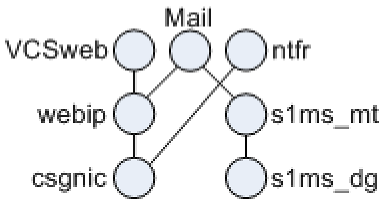
    The Veritas agent is also now installed on **Node_B**.

12. From the Veritas Cluster Server Cluster Manager, select **ImportTypes** from the **File** menu, which will display a file selection box.
13. Import the **MsgSrvTypes.cf** file from the **/etc/VRTSvcs/conf/config** directory.
14. Import this type file.

    You need to be on a cluster node to find this file.
15. Create a resource of type **MsgSrv** (for example, **Mail**).

    This resource requires the logical host name property to be set.
16. The Mail resource depends on **s1ms_mt** and **webip**. Create links between the resources as shown in Figure 10–5:

Figure 10–5 Veritas Cluster Dependencies (s1ms_mt and webip)



   a. Enable all resources and bring **Mail** online
   b. All servers should be started. Switch over to **Node_A** and check if the High Availability configuration is working.

MsgSrv Attributes and Arguments

This section describes MsgSvr additional attributes and arguments that govern the behavior of the mail resource.

Figure 10–4 describes the Veritas Server attributes.

Table 10–4 Veritas Server Attributes

Attribute	Description
FaultOnMonitorTimeouts	If unset (=0), monitor (probe) time outs are not treated as resource fault. Recommend setting this to 2. If the monitor times out twice, the resource will be restarted or failed over.

Table 10–4 (Cont.) Veritas Server Attributes

Attribute	Description
ConfInterval	Time interval over which faults/restarts are counted. Previous history is erased if the service remains online for this duration. Suggest 600 seconds.
ToleranceLimit	Number of times the monitor should return OFFLINE for declaring the resource FAULTED. Recommend leaving this value at 0 (default).

Figure 10–5 describes the MsgSvr attributes.

Table 10–5 Msg Svr Arguments

Option	Description
State	Indicates if the service is online or not in this system. This value is not changeable by the user.
LogHostName	The logical host name that is associated with this instance.
PrtStatus	If set to TRUE, the online status is printed to the Veritas Cluster Server log file.
DebugMode	If set to TRUE, the debugging information is sent to the Veritas Cluster Server log file.

To obtain the current values of following debug options:

```
 pwd
/opt/VRTSvcs/bin

hares -value ms-srvr DebugMode
hares -value ms-srvr PrtStatus
```

To set the following debug options:

```
 pwd
/opt/VRTSvcs/bin

hares -modify ms-srvr PrtStatus true
hares -modify ms-srvr DebugMode true
```

Unconfiguring High Availability

This section describes how to unconfigure high availability. To uninstall high availability, follow the instructions in your Veritas or Oracle Solaris Cluster documentation. The High Availability unconfiguration instructions differ depending on whether you are removing Veritas Cluster Server or Oracle Solaris Cluster.

To Unconfigure the Veritas Cluster Server

This section describes how to unconfigure the high availability components for Veritas Cluster Server.

1. Bring the **MAIL-RG** service group offline and disable its resources.
2. Remove the dependencies between the **mail** resource, the **logical_IP** resource, and the **mountshared** resource.
3. Bring the **MAIL-RG** service group back online so the **sharedg** resource is available.

4. Delete all of the Veritas Cluster Server resources created during installation.
5. Stop the Veritas Cluster Server and remove following files on both nodes:

```
/etc/VRTSvcs/conf/config/MsgSrvTypes.cf
/opt/VRTSvcs/bin/MsgSrv/online
/opt/VRTSvcs/bin/MsgSrv/offline
/opt/VRTSvcs/bin/MsgSrv/clean
/opt/VRTSvcs/bin/MsgSrv/monitor
/opt/VRTSvcs/bin/MsgSrv/sub.pl
```

6. Remove the Messaging Server entries from the **/etc/VRTSvcs/conf/config/main.cf** file on both nodes.
7. Remove the **/opt/VRTSvcs/bin/MsgSrv/** directory from both nodes.

Oracle Clusterware Installation and Configuration

Messaging Server can be configured with Oracle Clusterware. Be sure to review the Oracle Clusterware documentation prior to following these procedures.

To Install Oracle Clusterware

For information about installing Oracle Clusterware, see the overview of installing Oracle Clusterware in the *Oracle Clusterware Administration and Deployment Guide*.

To Configure Messaging Server to Use with Oracle Clusterware

After Oracle Clusterware is installed:

1. On each node in the cluster, including the NFS server (if used), create the Messaging Server runtime user and group under which the Messaging Server will run.

   The user ID and group ID numbers must be the same on all nodes in the cluster. The run time user ID is the user name under which Messaging Server runs. This name should not be root. The default is **mailsrv**. The runtime Group ID is the group under which Messaging Server runs. The default is **mail**. Although the configure utility can create these names for you, you can also create them before running configure as part of the preparation of each node as described in this chapter. The runtime user and group ID names must be in the following files:

   - **mailsrv**, or the name you select, must in **/etc/passwd** on all nodes in the cluster
   - **mail**, or the name you select, must be in **/etc/group** on all nodes in the cluster

2. Configure NFS shares with proper options on the NFS server machine and export them to the NFS clients (all Cluster nodes). Also make sure that NFS shares are on highly available storage. To use file systems other then NFS, like Failover File system or Clustered File systems created on shared storage, refer to Oracle Clusterware documentation.

   For NFS, mount all the NFS shares on all cluster nodes. These NFS mounts will be used for installing Messaging Server binaries, keeping configuration and runtime data.

   If a two-node symmetric cluster setup used, then two NFS mounts are needed on both nodes for Messaging Server instance 1 on node1 and Messaging Server instance 2 on node 2. Here is an example of the NFS share details from **/etc/exportfs** file on Linux.

```
Node 1 : /export/msg1 <NFS Client1 IP > (rw,nohide,insecure,no_subtree_
check,async,no_root_squash)

Node 2: /export/msg2 <NFS Client2 IP > (rw,nohide,insecure,no_subtree_
check,async,no_root_squash) - (in case of Symmetric HA)
```

On all cluster nodes, mount the NFS shares with following options. To make the mount points persistent across the boots, keep mount details in the **/etc/fstab** file.

```
Node 1:/export/msg1 /export/msg1 nfs
rw,bg,hard,intr,rsize=32768,wsize=32768,tcp,noac,vers=3,timeo=600

Node 2:/export/msg2 /export/msg2 nfs
rw,bg,hard,intr,rsize=32768,wsize=32768,tcp,noac,vers=3,timeo=600 (In case of
symmetric HA)
```

3. Create an HA logical host name resource and bring it on-line.

```
/u01/app/12.1.0/grid/bin $ ./appvipcfg create -network=1 -ip=<logical IP>
-vipname=<Logical IP Resource Name> -user=root
```

For example:

```
/u01/app/12.1.0/grid/bin $ ./appvipcfg create -network=1 -ip=10.0.0.3
-vipname=msg1 -user=root
```

4. Install the required Messaging Server packages on the primary node. Use the Messaging Server installer to install the Messaging Server packages.

   For asymmetric deployments: For NFS, install Messaging Server binaries on local file systems OR NFS mounts on each node of Oracle Clusterware. Install configuration data and run time data only on NFS mounted directory. For example, on **/export/msg1**.

   For symmetric deployments: For NFS, install Messaging Server binaries and configuration data on NFS mounts on node of Oracle Clusterware. For example, for Messaging Server instance 1 binaries and the configuration data on **/export/msg1** and Messaging Server instance 2 binaries and the configuration data on **/export/msg2**.

5. Configure the Messaging Server.

   In the initial runtime configuration, you are asked for the Fully Qualified Host Name. You must use the HA logical hostname instead of the physical hostname.

6. For legacy configuration, run the **ha_ip_config** script to set **service.listenaddr** and to configure the **dispatcher.cnf** and **job_controller.cnf** for high availability.

   The script ensures that the logical IP address is set for these parameters and files, rather than the physical IP address. It also enables the **watcher** process (sets **local.watcher.enable** to 1), and the auto restart process (**local.autorestart** to 1).

   The **ha_ip_config** script should only be run once on the primary node. **ha_ip_ config**is for legacy configuration. The corresponding Unified Configuration recipe is **msconfig run HAConfig.rcp**

7. Create an HA Messaging Server resource and start the resource.

   It is now time to create the HA Messaging Server resource. This resource is dependent upon the HA logical host name and HA disk resource if NFS mounts are not used (Eg. Cluster File system).

```
~ $ /u01/app/12.1.0/grid/bin/crsctl add type ocucs.ms.type -basetype cluster_
```

```
resource -attr "ATTRIBUTE=INSTANCE_PATH,TYPE=string,FLAGS=READONLY|REQUIRED"

~ $ /u01/app/12.1.0/grid/bin/crsctl add resource ocucs.ms.msg1 -type
ocucs.ms.type -attr " INSTANCE_PATH=<instance_path>/messaging64, AGENT_
FILENAME='%CRS_HOME%/bin/scriptagent', ACTION_SCRIPT='<instance_
path>/messaging64/cwha/bin/ms_actionscript.pl', ENABLED=1, AUTO_START=restore,
UPTIME_THRESHOLD=10m, CHECK_INTERVAL=10, SCRIPT_TIMEOUT=300, RESTART_
ATTEMPTS=2, OFFLINE_CHECK_INTERVAL=0, START_DEPENDENCIES='hard(msg1)
pullup(msg1)', STOP_DEPENDENCIES='hard(intermediate:msg1)',CARDINALITY=1,
FAILURE_INTERVAL=0, FAILURE_THRESHOLD=0, SERVER_POOLS=*,PLACEMENT=favored"

/u01/app/12.1.0/grid/bin $ /u01/app/12.1.0/grid/bin/crsctl stop resource msg1

/u01/app/12.1.0/grid/bin $ /u01/app/12.1.0/grid/bin/crsctl start resource
ocucs.ms.msg1 -n cl1
```

Where **ocucs.ms.type** is the Messaging Server resource type, **ocucs.ms.msg1** is the Messaging Server resource name, and **cl1** is the primary node.

For a Symmetric HA setup, you should create another Messaging Server resource on the primary node for the second Messaging Server installation.

To Unconfigure Oracle Clusterware

To unconfigure the high availability components for Oracle Clusterware.

1. Stop the Messaging server resource **ocucs.ms.msg1**.

   ```
   /u01/app/12.1.0/grid/bin/crsctl stop resource ocucs.ms.msg1
   ```

2. Remove the Messaging Server resource **ocucs.ms.msg1**.

   ```
   /u01/app/12.1.0/grid/bin/crsctl delete resource ocucs.ms.msg1
   ```

3. Stop the HA logical IP resource **msg1**.

   ```
   /u01/app/12.1.0/grid/bin/crsctl stop resource msg1
   ```

4. Remove the HA logical IP resource **msg1**.

   ```
   /u01/app/12.1.0/grid/bin/crsctl delete resource msg1
   ```

5. Remove the Messaging server resource type **ocucs.ms.type**.

   ```
   /u01/app/12.1.0/grid/bin/crsctl delete type ocucs.ms.type
   ```

6. Uninstall the Messaging Server.
7. Repeat the steps above for each instance in the Cluster setup if more then one instance is present.

Using the useconfig Utility

The **useconfig** utility allows you to share a single configuration between multiple nodes in an HA environment. This utility is not meant to upgrade or update an existing configuration. Note that only **useconfig** command usage has been changed in this release. All the MS HA info from the previous release is still valid.

For example, if you are upgrading your first node, you will install with the Installer and then configure Messaging Server. You will then failover to the second node where you will install the Messaging Server package with the Installer, but you will not have to run the Initial Runtime Configuration Program (**configure**) again. Instead, you can use the **useconfig** utility. To enable the utility, run **useconfig** to point to your previous Messaging Server configuration:

```
MessagingServer_home/sbin/useconfig MessagingServer_home/config
```

11

Configuring Messaging Server

This chapter provides information on how to perform an Oracle Communications Messaging Server initial configuration, as well as how to perform configurations for Messaging Server's individual components.

For information about performing a Cassandra message store initial configuration, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

High-level Overview of Configuring Messaging Server

Performing an initial run-time configuration of Messaging Server involves the following high-level steps:

1. Creating a UNIX system user and group for Messaging Server
2. Checking that DNS is properly configured
3. Preparing Directory Server for Messaging Server configuration by running the **comm_dssetup.pl** script
4. Creating the initial Messaging Server runtime configuration by running the **configure** command

Additionally, other steps to perform include the following:

1. Installing tools to provision Messaging Server
2. Modifying SMTP relay blocking configuration
3. Enabling Messaging Server startup after a reboot
4. Becoming familiar with best practices for performance tuning

Configuring Messaging Server

You must configure Messaging Server to complete the installation. You use the Messaging Server configuration command-line script, **configure**, to perform this initial runtime configuration. It is meant to create an initial runtime configuration to set up a generic functional Messaging Server configuration. Thus it gives you a base working configuration from which you can make your specific customizations. The program is only meant to be run once. Subsequent running of this program overwrites the existing configuration. To modify your initial runtime configuration, use the configuration utilities described here and in *Messaging Server System Administrator's Guide*.

When running **configure** on a Linux system that runs **portreserve**, it is possible that some ports that are commonly used by Messaging Server are pre-reserved. In that case, you may see a message like this during the configuration process:

```
WARNING: The following ports are currently reserved in /etc/portreserve
         143 (IMAP) is currently reserved by dovecot
         110 (POP) is currently reserved by dovecot
```

If you see such a message, you should investigate your proposed use of Messaging Server to see whether the ports listed would actually conflict. For example, if you are configuring an MTA relay only, the IMAP and POP port conflicts may be irrelevant because those ports would not be used by Messaging Server.

If you receive this warning, you should also investigate your operating system configuration to resolve the conflict including potentially disabling the reported services, if appropriate.

The **configure** command detects mismatches in certain critical LDAP attributes when performing second and subsequent initial configurations using the same LDAP server. The critical attributes are:

- default domain: **inetDomainBaseDN**, **preferredMailHost**, and s**unPreferredDomain**
- admin user: **userPassword**, **mailHost**, and **mail**

The admin's **userPassword** must match unless the **--novalidate** or **--noldap** options are used with **configure** (in which case the new value will replace the old one when the LDIF generated by **configure** is applied). In interactive mode, the admin may select whether to preserve or replace the other attributes. The default behavior is replace (as with previous versions), but the new **--preserveCritical** option changes the default behavior to preserve. If a state file is used, the default behavior is applied to all attributes except **userPassword**.

The initial runtime configuration must be completed for each Messaging Server components. Once this has been completed, you can configure Messaging Server's individual components.

The tasks to configure Messaging Server are as follows:

- Running the Messaging Server Initial Configuration Script
- Configuring the Message Transfer Agent (MTA)
- Configuring the Messaging Multiplexor (MMP)
- Configuring the Webmail Server

Running the Messaging Server Initial Configuration Script

You must run the Messaging Server initial configuration script before completing the installation. Follow the steps below to run the configuration script.

To Run the Messaging Server Configure Script:

1. Log in as or become the superuser (**root**).
2. Invoke the Messaging Server initial runtime **configure** command.

   It is recommend to use Unified Configuration for new Messaging Server deployments. For more information on Unified Configuration, see *Messaging Server System Administrator's Guide*.

For information on options you can set with your **configure** program, see Table A–1, " configure Options".

After running the **configure** command, the welcome text appears.

3. Select the directory where you want to store the Messaging Server configuration and data files.

   Symbolic links are created under the *MessagingServer_home* directory to the configuration and data directory. For more information on these symbolic links, see "Post-Installation Directory Layout".

   Make sure you have large enough disk space set aside for these files.

   The “Overwrite the existing configuration” prompt appears if you have an existing configuration.

   a. If you do receive the “Overwrite” message, to accept the default of yes, press Enter.
   b. Otherwise, type **n** to enter a different directory path.

4. Select the user name for server processes.

   To accept the default user name **mailsrv**, press Enter. Otherwise, type the user name for the server processes.

5. Select the group name for server processes.

   To accept the default group name mail, press Enter. Otherwise, type the group name for the server processes. This question appears only if the UNIX user name has not yet been created.

6. Select the fully-qualified local host name.

   This is the machine on which Messaging Server runs. When you installed the server, you might have specified the physical host name. However, if you are installing a cluster environment, use the logical host name. Here is the chance to change what you originally specified.

7. Type the default mail domain.
8. Select the host name for the LDAP Directory Server.
9. Select the LDAP administrator login.

   The Directory Manager has overall administrator privileges on the Directory Server and all servers (for example, Messaging Server) that make use of the Directory Server, and has full administration access to all entries in the Directory Server. The default and recommended Distinguished Name (DN) is **cn=Directory Manager** and is set during Directory Server configuration.

   If you are installing against a replicated Directory Server instance, you must specify the credentials of the replica, not the master directory.

10. Type the LDAP administrator password.

    Messages similar to the following appear:

    ```
    ==Checking Directory Server Setup from comm_dssetup
    Domain Suffix: o=isp
    User/Group Suffix: o=isp
    Mail List User Suffix: o=mlusers
    Schema Type: 2
    ```

11. Type a mail address for postmaster notices.

Select an address that your administrator actively monitors. For example, **pma@example.com** for a postmaster on the **example** domain. This address cannot begin with "Postmaster."

> **Note:** The user of the email address is not automatically created (although the default "admin" user is automatically created). Therefore, you need create it later by using a provisioning tool.

12. Type the IP addresses of hosts that are permitted to relay mail without authentication.

    You can use the **$(IP-pattern/significant-prefix-bits)** syntax. This information creates the appropriate mapping entries. It is important that you modify your configuration to match the needs of your site. Specifically, your Messaging Server should recognize its own internal systems and subnets from which SMTP relaying should always be accepted. If you do not update this configuration, you might encounter problems when testing your MTA configuration. For more information, see "Configuring SMTP Relay Blocking".

13. Type the password for administrator accounts.

    Type an initial password to be used for service administrator, server, user/group administrator, end user administrator privileges as well as PAB administrator and SSL passwords.

    After creating the initial runtime configuration, you might change this password for individual administrator accounts. For more information, see the discussion on how to modify your passwords in the *Messaging Server System Administrator's Guide*.

14. Verify the password for administration.
15. Retype the administration password.
16. The program displays the changes that it makes as well post-configuration changes that you might want to make.

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

Configuring the Message Transfer Agent (MTA)

Once you have followed the steps in "Running the Messaging Server Initial Configuration Script," you can finish configuring Messaging Server's MTA component.

To configure the Message Transfer Agent (MTA):

1. Disable the webmail server and message store.

    ```
    msconfig set store.enable 0
    msconfig set http.enable 0
    ```

2. Configure the relay for the kind of traffic you are dealing with and the kind of traffic shaping you need.

    For example, if your inbound relay needs to use LMTP, configure your deployment accordingly.

Configuring the Messaging Multiplexor (MMP)

Once you have followed the steps in "Running the Messaging Server Initial Configuration Script," you can finish configuring Messaging Server's Messaging Multiplexor (MMP) component.

To configure the Messaging Multiplexor (MMP) and disable other product components:

```
msconfig set mmp.enable 1
msconfig set store.enable 0
msconfig set mta.enable 0
msconfig set http.enable 0
```

Configuring the Webmail Server

Once you have followed the steps in "Running the Messaging Server Initial Configuration Script," you can finish configuring Messaging Server's Webmail Server component.

To Configure the Webmail Server:

1. Disable the message store and MTA on the webmail server host.

   ```
   msconfig set store.enable 0
   msconfig set mta.enable 0
   ```

2. (Optional) Set the following options.

   If you want to use a different store administrator or a non-standard IMAP port, use the options Table 11–1 for the back-end IMAP server(s):

Table 11–1 Configuration Options for Back-End IMAP Server(s)

Unified Configuration Option	Description
base.proxyadminpass	Default store administrator password. (Restart of HTTP service required and restart of IMAP service required.) Syntax: **string** Default: admin.password
base.proxyimapport	Default IMAP port number for backend store servers. (Restart of HTTP service required and restart of IMAP service required.) Syntax: **integer** Default: 143
base.proxyadmin	Default back-end store administrator login name. (Restart of HTTP service required and restart of IMAP service required.) Syntax: **string** Default: admin

The Webmail Server can communicate with multiple back-end IMAP servers. If the IMAP servers use different values for these options, you must set individual values for each host, as shown in Table 11–2:

Table 11–2 Individual Configuration Values for Hosts

Unified Configuration Option	Description
proxy:*hostname***.admin** **proxy:***hostname***.adminpass** **proxy.***hostname***.imapport**	Controls aspects of proxy connection authentication and port and hosts. Note that since such options are set under a named proxy group, where the group name is a host name.

where *hostname* is the name of the host on which each back-end IMAP server is running.

Note: In general in Unified Configuration, for proxy-related options there should be two scopes for the same option:

- **base.***option* is the global scope.
- **proxy:***hostname.option* is the host-specific scope.

Currently, an error in Unified Configuration causes the same option to have two different names depending on the scope. Thus, **base.proxyimapport** is equivalent to **proxy:***hostname***.imapport**, **base.proxyimapport** is equivalent to **proxy:***hostname***.imapport**, and **base.proxyimapadminpass** is equivalent to **proxy:***hostname***.imapadminpass**. In addition, there is no host-specific form for **base.proxyimapssl**. It is a single global setting.

Configuring Webmail Server Examples

For one back-end IMAP server:

```
msconfig set base.proxyadmin -myadmin
msconfig set base.proxyadminpass password
msconfig set base.proxyimapport -143
```

Configuring Oracle Communications Messaging Server Individual Ports

When multiple instances of the Messaging Server are installed on one host, they are initially configured to use the same ports. If you run both instances of the product simultaneously, the ports conflict.

To avoid conflicts, configure the ports for additional instances of the product so that they differ. (The initial instance can retain the original port settings as long as other instances are modified so that port numbers aren't re-used.)

Messaging Server can change the ports for the following processes:

- SMTP
- IMAP
- IMAP SSL
- POP
- POP over SSL
- HTTPD
- ENS
- job_controller

- watcher

The SSL versions of the ports must also be unique between instances. There may be additional ports to reconfigure that are not listed here, such as SMTP SUBMIT.

To look for MTA-related processes, you can use the following techniques:

- In Unified Configuration:

```
msconfig
msconfig > show *port*
```

In addition, you can identify ports by taking these actions:

- See Table 9–1, " Messaging Server Default Ports" for information on Default Port Numbers.
- See configuring POP, IMAP, and HTTP services in the *Messaging Server System Administrator's Guide*.
- You can **grep** for the word "port" in the **masterconfig** file (**lib/config.meta**).
- Query the following options as shown in Table 11–3:

Table 11–3 Port Options

Service	Unified Configuration Option	Legacy Configuration configutil Option	Default Value
watcher	**watcher.port**	**local.watcher.port**	49994
metermaid	**metermaid.port**	**metermaid.config.port**	63837
IMAP	**imap.port**	**service.imap.port**	143
IMAP SSL	**imap.sslport**	**service.imap.sslport**	993
POP	**pop.port**	**service.pop.port**	110
POP over SSL	**pop.sslport**	**service.pop.sslport**	995
Webmail	**http.port**	**service.http.port**	80
Webmail SSL	**http.sslport**	**service.http.sslport**	443
ens	**notifytarget:name.ensport**	**local.store.notifyplugin.ensport**	7997
jmq	**notifytarget:name.jmqport**	**local.store.notifyplugin.jmqport**	7676

Configuring an Oracle Communications Messaging Server Host to be Multi-Homed

When you install multiple instances of Messaging Server on the same host, the different product instances use the same ports. If you run both instances of Messaging Server simultaneously, the ports conflict.

This information describes how to sue a different IP address for each installation and configure the host to be multi-homed (accepting multiple IP addresses).

This section includes the following topics:

- To Change the IP Address for Each Installation
- To Configure the Host to be Multi-Homed
- To Configure Multiple Addresses Per Interface
- Multi-Home Example

To Change the IP Address for Each Installation

- Run the **ha_ip_config** utility. You must configure each installation to use a specific IP address, since the out-of-the-box default is to respond to any IP address (**INADDR_ANY**).

> **Note:** The ENS service needs a separate step to change the IP address it responds to. A workaround is to either disable the ENS server for one of the installations (Unified Configuration uses **ens.enable**, legacy configuration uses **local.ens.enable**), or to change the port used by the ENS server. If you don't do this, one of the ENS servers does not start up. This may not be a huge issue at this time because the other ENS server handles the requests.

To Configure the Host to be Multi-Homed

To configure a host to be multi-homed, see the Oracle Solaris documentation.

To Configure Multiple Addresses Per Interface

To configure multiple addresses per interface, see the Oracle Solaris documentation.

Multi-Home Example

The following example creates a multi-home on the host **myhost**.

1. Create the new interface:

```
ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
index 1
        inet 127.0.0.1 netmask ff000000
e1000g0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.1.110.114 netmask ffffff80 broadcast 10.1.110.127
        ether 0:c:f1:8e:fb:4
ifconfig  e1000g0:1 plumb
ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
index 1
        inet 127.0.0.1 netmask ff000000
e1000g0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.1.110.114 netmask ffffff80 broadcast 10.1.110.127
        ether 0:c:f1:8e:fb:4
e1000g0:1: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 0.0.0.0 netmask 0
ifconfig e1000g0:1 10.1.110.16 up
```

2. Set the IP address for the Messaging Server on the alternate root (on **/var/tmp/altroot/opt/sun/comms/messaging64** in the following examples):

   Unified Configuration example:

```
cd /var/tmp/altroot/opt/sun/comms/messaging64/sbin
msconfig run /opt/sun/comms/messaging64/lib/recipes/HAConfig.rcp

Logical HA IP address of the cluster: 10.1.110.16
```

   Legacy configuration example:

```
cd /var/tmp/altroot/opt/sun/comms/messaging64/sbin
```

```
ha_ip_config

Please specify the IP address assigned to the HA logical host name. Use
dotted decimal form, a.b.c.d

Logical IP address: 10.1.110.16

Please specify the path to the top level directory in which Messaging Server
is
installed.

Messaging Server server root: /var/tmp/altroot/opt/sun/comms/messaging64

The Messaging Server server root directory does not contain any slapd-*
subdirectories.
Skipping configuration of LDAP servers.

        Logical IP address: 10.1.110.16
        Messaging Server server root:
/var/tmp/altroot/opt/sun/comms/messaging64

Do you wish to change any of the above choices (yes/no) [no]?

Updating the file
/var/tmp/altroot/opt/sun/comms/messaging64/config/dispatcher.cnf
Updating the file /var/tmp/altroot/opt/sun/comms/messaging64/config/job_
controller.cnf
Setting the service.listenaddr configutil option
Setting the service.http.smtphost configutil option
Setting the local.watcher.enable configutil option
Setting the local.autorestart configutil option
Setting the metermaid.config.listenaddr configutil option
Setting the metermaid.config.serverhost configutil option
Setting the local.ens.port configutil option
Configuration successfully updated
```

3. Do the same for the Messaging Server:

   Unified Configuration:

```
cd /opt/sun/comms/messaging64/sbin
ha_ip_config

Please specify the IP address assigned to the HA logical host name. Use
dotted decimal form, a.b.c.d

Logical IP address: 10.1.110.114

Please specify the path to the top level directory in which Messaging Server
is
installed.

Messaging Server server root: /opt/sun/comms/messaging64

The Messaging Server server root directory does not contain any slapd-*
subdirectories.
Skipping configuration of LDAP servers.

        Logical IP address: 10.1.110.114
        Messaging Server server root: /opt/sun/comms/messaging64
```

```
Do you wish to change any of the above choices (yes/no) [no]?

Updating the file /opt/sun/comms/messaging64/config/dispatcher.cnf
Updating the file /opt/sun/comms/messaging64/config/job_controller.cnf
Setting the base.listenaddr msconfig option
Setting the http.smtphost msconfig option
Setting the watcher.enable msconfig option
Setting the base.autorestart.enable msconfig option
Setting the metermaid.listenaddr msconfig option
Setting the metermaid_client.server_host msconfig option
Setting the ens.port msconfig option
Configuration successfully updated
```

Legacy configuration:

```
cd /opt/sun/comms/messaging64/sbin
ha_ip_config

Please specify the IP address assigned to the HA logical host name. Use
dotted decimal form, a.b.c.d

Logical IP address: 10.1.110.114

Please specify the path to the top level directory in which Messaging Server
is
installed.

Messaging Server server root: /opt/sun/comms/messaging64

The Messaging Server server root directory does not contain any slapd-*
subdirectories.
Skipping configuration of LDAP servers.

        Logical IP address: 10.1.110.114
        Messaging Server server root: /opt/sun/comms/messaging64

Do you wish to change any of the above choices (yes/no) [no]?

Updating the file /opt/sun/comms/messaging64/config/dispatcher.cnf
Updating the file /opt/sun/comms/messaging64/config/job_controller.cnf
Setting the service.listenaddr configutil parameter
Setting the service.http.smtphost configutil option
Setting the local.watcher.enable configutil option
Setting the local.autorestart configutil option
Setting the metermaid.config.listenaddr configutil options
Setting the metermaid.config.serverhost configutil options
Setting the local.ens.port configutil option
Configuration successfully updated
```

4. Disable the ENS Server on one of the installation by setting **ens.enable** (Unified Configuration) or **local.ens.enable** (legacy configuration) to 0:

   Unified Configuration:

   ```
   msconfig -o ens.enable -v 0
   ```

   Legacy configuration:

   ```
   configutil -o local.ens.enable -v 0
   ```

5. Configure the netmask and broadcast on the new IP address:

```
ifconfig e1000g0:1 down
ifconfig e1000g0:1 netmask 0xffffff80
ifconfig e1000g0:1
e1000g0:1: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.1.110.16 netmask ffffff80 broadcast 10.255.255.255
ifconfig e1000g0:1 broadcast 10.1.110.127
ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
index 1
        inet 127.0.0.1 netmask ff000000
e1000g0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.1.110.114 netmask ffffff80 broadcast 10.1.110.127
        ether 0:c:f1:8e:fb:4
e1000g0:1: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.1.110.16 netmask ffffff80 broadcast 10.1.110.127
ifconfig e1000g0:1 up
```

6. Edit **/etc/hosts** to add the new IP address 10.1.110.16 to it:

```
cat /etc/hosts
127.0.0.1       localhost
10.1.110.114    myhost.west.example.com myhost        loghost
10.1.110.4      elegit.west.example.com
multi-home - second IP address on ethernet port
10.1.110.16     myhost2.west.example.com myhost2
```

12 Messaging Server Post-Installation Tasks

This chapter describes the post-installation tasks that you must complete to finish the Oracle Communications Messaging Server installation.

For information about Cassandra message store post-installation tasks, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Installing Messaging Server Provisioning Tools

This information describes the schema and provisioning options for Messaging Server. Because of the complexity in provisioning Messaging Server, you need to understand your options before installing the product.

Understanding Messaging Server Schema Choices

Two schema options are available and supported with Messaging Server: LDAP Schema version 1 and LDAP Schema version 2.

Note: For information on how to migrate from Sun Java System LDAP Schema version 1 to Sun Java System LDAP Schema version 2, see:

`http://docs.oracle.com/cd/E19636-01/819-2656/index.html`

LDAP Schema 1 and Messaging Server

LDAP Schema 1 is a provisioning schema that consists of both an Organization Tree and a DC Tree. This set of schema (at the time, it was simply called "schema") was supported in earlier Messaging Server versions.

In Schema 1, when Messaging Server searches 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.

LDAP Schema 1 and Messaging Server Supported Provisioning Tools

Schema 1 is supported by both Sun ONE Delegated Administrator and Oracle Communications Delegated Administrator. For more information, see "Understanding Messaging Server Provisioning Tools".

LDAP Schema 2 (Native Mode) and Messaging Server

LDAP Schema 2 is a set of Organization nodes (each may serve one or more domain names) and users entries typically live beneath the Organization nodes.

Note: If you have an existing Messaging Server installation that uses Schema 1, and you want to integrate with other Communications Suite products, you should migrate your directory to Schema 2 after you upgrade Messaging Server. For information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2, see:

`http://docs.oracle.com/cd/E19636-01/819-2656/index.html`

LDAP Schema 2 and Messaging Server Supported Provisioning Tools

Schema 2 supports Delegated Administrator. For more information, see "Understanding Messaging Server Provisioning Tools".

LDAP Schema 2 Compatibility Mode and Messaging Server

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. For more information, see:

`http://docs.oracle.com/cd/E19636-01/819-2656/index.html`

Understanding Messaging Server Provisioning Tools

This section describes supported provisioning tools that enable you to query, modify, add, or delete user, group, and domain entry information in your LDAP directory.

Through supported Messaging Server provisioning tools, you can query, modify, add, or delete user, group, and domain entry information in your LDAP directory. This section examines these Messaging Server provisioning tools.

You should use Messaging Server Provisioning Mechanisms to evaluate your schema and provisioning tool options.

Note: Prior to installing and configuring Messaging Server, you need to decide upon a schema model and tool or tools for provisioning your Messaging Server entries.

The following sections provide high-level information about the supported provisioning tools:

- LDAP Provisioning Tools for Messaging Server
- Comparing Messaging Server Provisioning Tool Options

LDAP Provisioning Tools for Messaging Server

Schema 1 users and groups can be provisioned using the LDAP Directory tools (Schema 2 is not supported). Unlike the Delegated Administrator graphical and command-line interfaces, you can directly provision users and groups by adding, removing, and modifying the LDIF records through LDAP without having to use a user interface.

Comparing Messaging Server Provisioning Tool Options

Table 12–1 shows the various supported schema, provisioning tools, provisioning limitations, and recommended documentation for additional information.

Table 12–1 Messaging Server Provisioning Mechanisms

Supported Provisioning Tool	Provisioning Tool Functionality	Provisioning Tool Limitations	For Further Information
LDAP Provisioning Tools Uses: Schema 1	Provides tools to directly modify LDAP entries or for creating custom provisioning tools.	▪ Incompatible with Oracle Schema 2 and with other Java Enterprise System products.	Read the *Schema Reference*. Describes the LDAP Schema 1 provisioning model. In addition, this guide explain how to use LDAP provisioning tools and the usage of specific attributes and object classes.
Delegated Administrator Uses: Schema 2	Provides graphical and command-line interfaces for administrators to manage users, groups, domains, and mailing lists. Compatible with other Communications Suite products.	▪ Not applicable.	Read the *Delegated Administrator System Administrator's Guide*. Provides syntax and usage for the command-line utility.

Configuring SMTP Relay Blocking

The **configure** program prompts you to enter host IP addresses that are allowed as SMTP relay hosts. The **configure** program uses this information to construct the appropriate mapping entries.

By default, Messaging Server is configured to block attempted SMTP relays. That is, Messaging Server rejects attempted message submissions to external addresses from unauthenticated external sources (external systems are any other system than the host on which the server itself resides). This default configuration is quite aggressive in blocking SMTP relaying in that it considers all other systems to be external systems.

IMAP and POP clients that attempt to submit messages by using Messaging Server system's SMTP server destined for external addresses, and which do not authenticate using SMTP AUTH (SASL), find their submission attempts rejected. Which systems and subnets are recognized as internal is typically controlled by the **INTERNAL_IP** mapping table. In Unified Configuration, this mapping table is part of the overall configuration, and is viewed or edited by using the **msconfig** command. In legacy configuration, this mapping table is found in the *MessagingServer_home*/**config/mapping** file.

For instance, on a Messaging Server system whose IP address is **192.45.67.89**, the default **INTERNAL_IP** mapping table would appear as follows:

```
INTERNAL_IP
```

```
  $(192.45.67.89/32) $Y
  127.0.0.1 $Y
  * $N
```

The initial entry, using the **S (IP-pattern/significant-prefix-bits)** syntax, is specifying that any IP address that matches the full 32 bits of **192.45.67.89** should match and be considered internal. the second entry recognizes the loopback IP address **127.0.0.1** as internal. The final entry specifies that all other IP addresses should not be considered internal.

You can add additional entries by specifying additional IP addresses or subnets before the final **$N** entry. These entries must specify an IP address or subnet (using the **$(.../...)** syntax to specify a subnet) on the left side and **$Y** on the right side. Or you can modify the existing **$(.../...)** entry to accept a more general subnet.

For instance, if this same sample site has a class C network, that is, it owns all of the **192.45.67.0** subnet, then the site would want to modify the initial entry so that the mapping table appears as follows:

```
INTERNAL_IP
$ (192.45.67.0/24) $Y
127.0.0.1 SY
* $N
```

Or if the site owns only those IP addresses in the range **192.45.67.80-192.45.67.99**, then the site would want to use:

```
INTERNAL_IP
! Match IP addresses in the range 192.45.67.80-192.45.67.95
 $ (192.45.67.80/28) $Y
! Match IP addresses in the range 192.45.67.96-192.45.67.99
 $ (192.45.67.96/30) $Y
 127.0.0.1 $Y
 * $N
```

The *MessagingServer_home*/**bin/imsimta test -match** utility can be useful for checking whether an IP address matches a particular **$(.../...)** test condition. The **imsimta test -mapping** utility can be more generally useful in checking that your **INTERNAL_IP** mapping table returns the desired results for various IP address inputs.

After modifying your **INTERNAL_IP** mapping table, be sure to issue the *MessagingServer_home*/**bin/imsimta cnbuild** (if you are using a compiled configuration) and the *MessagingServer_home*/**bin/imsimta restart** utilities so that the changes take effect.

Further information on the mapping file and general mapping table format, as well as information on **imsimta** command line utilities, can be found in the *Messaging Server System Administrator's Guide.* In addition, information on the **INTERNAL_IP** mapping table can be found in the *Messaging Server System Administrator's Guide.*

Using Service Management Framework with Messaging Server

SMF support has been integrated into the product. Messaging Server provides a single SMF service definition file.

SMF was added in Solaris OS 10 as a replacement to the **/etc/init.d** scripts for starting, stopping, and restarting services. SMF dramatically decreases boot time as it is aware of dependencies between services, and starts services in parallel where possible.

```
MessagingServer_home/data/install/messaging.xml
```

The SMF service definitions can be imported using the **svccfg** command.

```
svccfg import DataRoot/install/messaging.xml
```

The following example shows how to check initial Messaging Server status, enable SMF, then verify status. Please note that Messaging Server must be stopped prior to using the **svcadm enable** command.

```
svcs messaging_server

STATE          STIME    FMRI
disabled        8:58:29 svc:/network/messaging_server:default

svcadm enable messaging_server

svcs messaging_server
STATE          STIME    FMRI
online          9:08:54 svc:/network/messaging_server:default
```

For more information on SMF, see the overview about Managing Services in the *Solaris System Administration Guide*. This guide provides an overview of SMF, including SMF concepts, administrative and programming interfaces, components, and run levels.

Enabling Startup After a Reboot

You can enable Messaging Server startup after system reboots by using the bootup script. On Linux, this script is *MessagingServer_home*/**data/install/Sun_MsgSvr**. For Solaris OS 10, you should use the Service Management Framework. That is, by default, Messaging Server is not restarted after a system reboot unless you run this script. In addition, this script can also start up your MMP, if enabled.

To Enable Messaging Server After a Reboot on Solaris

1. Copy the *MessagingServer_home* **/data/install/Sun_MsgSvr** script into the **/etc/init.d** directory.
2. Change the following ownerships and access modes of the **Sun_MsgSvr** script:

Table 12–2 Ownerships and Access Modes of Sun_MsgSvr Script

Ownership (chown(1M))	Group Ownership (chgrp(1M))	Access Mode (chmod(1M))
root (superuser)	**sys**	0744

3. Change directories to the **/ect/rc2.d** directory and create the following link:

   ```
   ln /etc/init.d/Sun_MsgSvr S92Sun_MsgSvr
   ```

4. Change directories to the **/ect/rc0.d** directory and create the following link:

   ```
   ln /etc/init.d/Sun_MsgSvr K08Sun_MsgSvr
   ```

To Enable Messaging Server After a Reboot on Linux

Products that Messaging Server uses need to be started in a specific order to ensure that any pre-requisite services are enabled prior to the product starting. This ordering is especially important when starting the products when booting the server.

The ordering of the product start-up is as follows:

When the server is shut-down, the ordering is (roughly) reversed.

1. Directory Server (relies on network interface availability)
2. JMQ (for Messaging Server notifications)
3. Messaging Server (relies on Directory Server for user-group information)

Oracle Linux and Red Hat Enterprise Linux provide the **chkconfig** utility to control the ordering of the product start-up and shut-down. A good explanation of the Red Hat Linux **chkconfig** functionality is available here:

`http://www.linuxjournal.com/article/4445`

The logs of each product being started/stopped during of the boot-up and shut-down is stored in **/var/log/boot.log** file on the server.

Performance and Tuning

For information on performance and tuning considerations for Messaging Server, see "Performance Tuning Considerations for a Messaging Server Architecture".

Post-Installation Directory Layout

After installing Messaging Server, its directories and files are arranged in the organization described in Table 12–3. The table shows only those directories and files of most interest for typical server administration tasks.

Table 12–3 Post-Installation Directories and Files

Directory	Default Location and Description
Messaging Server Base *MessagingServer_home*	**/opt/sun/comms/messaging64** (default location) The directory on the Messaging Server machine dedicated to holding the server program, configuration, maintenance, and information files. To configure more than one Messaging Server base directory per machine, see the discussion on the ALTROOT command-line argument in "commpkg Reference".
Configuration **config**	*MessagingServer_home***/config/** Contains all of the Messaging Server configuration files, such as **config.xml** for Unified Configuration, or the **imta.cnf** and the **msg.conf** files, for legacy configuration. This directory is symbolically linked to the **config** subdirectory of the data and configuration directory (default: **/var/opt/sun/comms/messaging64/**) that you specified in the initial runtime configuration.
Log **log**	*DataRoot***/log/** A convenience symbolic link to *DataRoot***/log**, which contains the Messaging Server log files like the **mail.log_current** file.

Table 12–3 (Cont.) Post-Installation Directories and Files

Directory	Default Location and Description
Data **data**	*DataRoot* Contains databases, configuration, log files, site-programs, queues, store and message files. The data directory includes the **config** and **log** directories. This directory is by default symbolically linked (on UNIX platforms) to the data and configuration directory (default: **/var/opt/sun/comms/messaging64**) that you specified in the initial runtime configuration.
System Administrator Programs **bin**	*MessagingServer_home*/**bin/** Contains the Messaging Server system administrator executable programs and scripts such as **imsimta**, **msconfig**, **configutil**, **stop-msg**, **start-msg**, and **uninstaller**.
Library **lib**	*MessagingServer_home*/**lib/** Contains shared libraries, private executable programs and scripts, daemons, and non-customizable content data files. For example: **imapd** and **qm_maint.hlp.**
SDK Include Files **include**	*MessagingServer_home*/**include/** Contains Messaging header files for Software Development Kits (SDK).
Examples **examples**	*MessagingServer_home*/**examples/** Contains the examples for various SDKs.
Installation Data **install**	*MessagingServer_home*/**data/install/** and *MessagingServer_home*/**data/setup/** Contains installation-related data files such as installation log files, silent installation files, factory default configuration files, and the initial runtime configuration log files.

Post-Installation Port Numbers

In the installation and initial runtime configuration programs, port numbers are chosen for various services. These port numbers can range from 1 to 65535. Select numbers that do not conflict with port numbers used by enabled system services or other third-party software. The authoritative list of registered port numbers is available at `http://www.iana.org`. The **/ect/services** also lists a subset of these numbers.

Table 12–4 lists the port numbers that are designated after installation.

Table 12–4 Port Numbers Designated During Installation

Service	Port	Unified Configuration Option to Change Port	Legacy Configuration Option to Change Port	Unified Configuration Option to Enable/Disable Service	Legacy Configuration Option to Enable/Disable Service
Message Store	NA	NA	NA	**store.enable** (1)	**local.store.enable** (1)
IMAP Server	143	**imap.port**	**service.imap.port**	**imap.enable** (1)	**service.imap.enable** (1)
POP Server	110	**pop.port**	**service.pop.port**	**pop.enable** (1)	**service.pop.enable** (1)

Table 12–4 (Cont.) Port Numbers Designated During Installation

Service	Port	Unified Configuration Option to Change Port	Legacy Configuration Option to Change Port	Unified Configuration Option to Enable/Disable Service	Legacy Configuration Option to Enable/Disable Service
IMAPS Server	993	**imap.sslport**	**service.imap.sslport**	**imap.enablesslport** (0)	**service.imap.enablesslport** (0)
POPS Server	995	**pop.sslport**	**service.pop.sslport**	**pop.enablesslport** (0)	**service.pop.enablesslport** (0)
LMTP Server	225	**dispatcher.service:LMTP.tcp_ports**	**dispatcher.cnf**	**dispatcher.service:LMTP.enable**	**dispatcher.cnf** (disabled)
MTA	NA	NA	NA	**mta.enable**	**local.imta.enable** (1)
SMTP Relay	25	**dispatcher.service:SMTP.tcp_ports**	**dispatcher.cnf**	**dispatcher.service:SMTP.enable**	**dispatcher.cnf** (enabled)
SMTP Submit	587	**dispatcher.service:SMTP_SUBMIT.tcp_ports**	**dispatcher.cnf**	**dispatcher.service:SMTP_SUBMIT.enable**	**dispatcher.cnf** (enabled)
SMTPS Submits	465	**dispatcher.service:SMTP_SUBMIT.tcp_ports**	**dispatcher.cnf**	**dispatcher.service:SMTP_SUBMIT.enable**	**dispatcher.cnf** (disabled)
http mail proxy	8990	**http.port**	**service.http.port**	**http.enable** (1)	**local.http.enable** (1)
https mail proxy	8991	**http.sslport**	**service.http.sslport**	**http.enablesslport** (0)	**service.http.enablesslport** (0)
MMP	NA	NA	NA	**mmp.enable** (0)	**local.mmp.enable** (0)
IMAP Proxy	143	**imapproxy.tcp_ listen:imapproxy1.tcp_ ports**	**Aservice.cfg**	NA	**Aservice.cfg** (0)
POP Proxy	110	**popproxy.tcp_ listen:poppro xy1.tcp_ports**	**Aservice.cfg**	NA	**Aservice.cfg** (0)
Submit Proxy	587	**submitproxy.tcp_ listen:poppro xy1.tcp_ports**	**Aservice.cfg**	NA	**Aservice.cfg** (0)
IMAPS Proxy	993	**proxyimapssl**	**Aservice.cfg** and **ImapProxyAService.cfg**	NA	**Aservice.cfg** and **ImapProxyAService.cfg** (disabled)
POPS Proxy	995	**popproxy.tcp_listen:ssl_ports**	**Aservice.cfg** and **PopProxyAService.cfg**	NA	**Aservice.cfg** and **PopProxyAService.cfg** (disabled)

Table 12–4 (Cont.) Port Numbers Designated During Installation

Service	Port	Unified Configuration Option to Change Port	Legacy Configuration Option to Change Port	Unified Configuration Option to Enable/Disable Service	Legacy Configuration Option to Enable/Disable Service
Submits Proxy	465	**submitproxy.tcplisten:ssl_ports**	**Aservice.cfg** and **SmtpProxyAService.cfg**	NA	**Aservice.cfg** and **SmtpProxyAService.cfg** (0)
Internal Servers	NA	NA	NA	NA	NA
watcher	49994	**watcher.port**	**local.watcher.port**	**watcher.enable** (1)	**local.watcher.enable** (1)
job_ controller	27442	**job_ controller.tcp_ports**	**job_ controller.cnf**	**mta.enable** (1)	**local.imta.enable** (1)
ENS	7997	**ens.port**	**local.ens.port**	**ens.enable** (0)	**local.ens.enable** (0)

JMQ Notification

Messaging Server can use Oracle GlassFish Message Queue, a standards-based messaging service, to send event notifications. Message Queue is provided as a shared component when you install Messaging Server or other Communications Suite products.

For more information, see the discussion on integrating JMQ and Messaging Server in the *Messaging Server System Administrator's Guide.*

Configuring Certificate Based Authentication

Messaging Server supports client certificate authentication. For more information, see the discussion on certificate based authentication for Messaging Server in the *Messaging Server Security Guide.*

13

Upgrading Messaging Server

This chapter describes the three Messaging Server upgrade strategies and procedures to upgrade from Messaging Server 7.x, 8.0, or 8.0.1, to Messaging Server 8.0.2. It assumes that you have chosen a target deployment, and have developed an architectural design and deployment plan.

About Upgrading to Messaging Server 8.0.2

Messaging Server 8.0.2 provides a redesigned message store that is based on DataStax Cassandra Database. However, you cannot upgrade to the Cassandra message store. You must perform a migration. To migrate mailboxes from classic message store to Cassandra message store, you must perform a number of steps, including installing message access layer hosts, configuring store affinity groups, installing Cassandra database, rehosting users, and so on. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Note: You cannot install both classic message store and Cassandra message store in the same Messaging Server deployment.

You can upgrade Messaging Server 7.x, 8.0, and 8.0.1 back-end message stores, MTA, MMP, and **mshttp** components to Messaging Server 8.0.2.

Note: Prior to this release, once you upgraded to Oracle Communications Messaging Server 7.0.5 or greater, from a version prior to Messaging Server 7.0.5, you could not downgrade by ”backing out” the upgrade. This was because of database incompatibilities with prior versions starting in Messaging Server 7.0.5. This situation does not apply Messaging Server 8.0.2 and Cassandra message store, as you cannot upgrade a Messaging Server 7.x, 8.0, or 8.01 classic message store to 8.0.2 and Cassandra message store; and so, the previous "backing out" scenario no longer applies in that situation.

Upgrade Requirements for Messaging Server

The requirements for upgrading to Messaging Server 8.0.2 are:

- You must be running at least Messaging Server 7.x to upgrade to Messaging Server 8.0.2.

- You cannot upgrade from Messaging Server 5.x or 6.x directly to Messaging Server 8.0.2. You first must upgrade to Messaging Server 7.x, then upgrade to Messaging Server 8.0.2.

  Contact Oracle Consulting to upgrade directly from Messaging Server 5.x or 6.x to Messaging Server 8.0.2.
- Linux platforms: Messaging Server supports Oracle Linux/ Red Hat Enterprise. See "Supported Operating Systems" for version information.

Note: This document uses the side-by-side installation method to be consistent between Solaris and Linux platforms. In general, you should avoid using the alternate root method when upgrading Messaging Server, because Solaris now uses alternative root for its Live Upgrade feature.

Upgrade Features

This Messaging Server upgrade includes the following features, which simplify the side-by-side upgrade method:

- Upgrade Does Not Touch Messaging Server Data or Configuration
- Improvements to the stored -r Command
- Solaris SRV4 Patches

Upgrade Does Not Touch Messaging Server Data or Configuration

Messaging Server package scripts and **preupgrade** and **postupgrade** scripts no longer alter the data and configuration in any way. In addition, the upgrade no longer automatically runs the **stop-msg** command when uninstalling.

For side-by-side migrations, this feature enables you to install two separate Messaging Server versions, such as 7.0.5 and 8.0.2, on the same host, that point to the same data and configuration, and activate a version by running that version's specific **start-msg** command. The Messaging Server data and configuration are “upgraded” when the **start-msg** script invokes the **updateCfgVersion** script after detecting that a new Messaging Server version is used for the first time.

Note: When you upgrade from a version of Messaging Server prior to 7.0.5, the **restricted.cnf file** is not created until you start Messaging Server services after the upgrade. If you attempt to run any Messaging Server commands before starting Messaging Server, you receive an error message that the **restricted.cnf** file does not exist. Thus, if you want to make configuration changes after upgrading, and before starting Messaging Server services, you must first manually run the **updateCfgVersion** script.

Improvements to the stored -r Command

Messaging Server upgrade no longer runs the **stored -r** command prior to uninstalling the previous version's binaries.

Solaris SRV4 Patches

Messaging Server SVR4 style patches are no longer available on Solaris. Instead, you use Automated Release Update (ARU) patches. ARU patches treat each Messaging Server release and subsequent versions as a different package version. For example, Messaging Server 8.0.2 would have a different package version than Messaging Server 8.0.3. Because of this versioning, you can install two copies of the same version of Messaging Server on the same host. Thus, for upgrades, you no longer need to use the alternate root (**ALTROOT**) install method.

About Messaging Server Unified Configuration

Beginning with Messaging Server 7 Update 5, Messaging Server has the capability to create a Unified Configuration. Unified Configuration provides an improved, streamlined process to configure and administer Messaging Server. Unlike in legacy configurations (Messaging Server 7 Update 4 and prior releases), Unified Configuration uses validation to verify configuration accuracy, and employs a single tool to configure the entire Messaging Server configuration (with a few exceptions). Thus, moving your deployment to Unified Configuration simplifies administration and reduces configuration mistakes.

After upgrading to Messaging Server 7 Update 5 and later, you can decide to migrate your legacy configuration to Unified Configuration. When you convert to Unified Configuration, Messaging Server saves your old legacy configuration in the *ConfigRoot*/**legacy-config** directory. If necessary, you can restore a saved legacy configuration at the time you converted, however, all changes made to your configuration after converting to United Configuration are lost. You can migrate to Unified Configuration after you have completed the upgrade. You are not required to migrate to Unified Configuration during the upgrade process.

To help you decide to migrate to Unified Configuration, see *Messaging Server System Administrator's Guide* for more information about Unified Configuration capabilities and tools.

Upgrading Messaging Server Overview

A Messaging Server deployment can consist of multiple back-end message stores, multiple webmail servers, front-end MMPs, and MTA relays. Like all upgrades, you proceed on a host-by-host basis.

Note: If you want to use the Cassandra message store in Messaging Server 8.0.2, you must perform a migration of the classic message store to Cassandra messages store. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Upgrading a Messaging Server deployment includes the following high-level steps:

1. Backing up the Messaging Server data
2. Upgrading and running **comm_dssetup.pl** to the latest version before upgrading Messaging Server

   Messaging Server 8.0.2 requires you to apply at least **comm_dssetup.pl** version 6.4.0.28.0 against Directory Server. The Messaging Server 8.0.2 media pack includes **comm_dssetup.pl** version 6.4.0.28.0.

3. Defining your upgrade target and the required products and components for that target
4. Reviewing your Messaging Server architecture and topology

   Although you might be satisfied with your current Messaging Server architecture and topology, upgrading can provide the opportunity to redesign your deployment for more optimal performance. See "Developing a Messaging Server Architecture" and "Planning a Messaging Server Sizing Strategy" for more information.
5. Selecting the upgrade sequence of individual Messaging Server hosts

   This includes upgrading components such as message store servers, proxies, webmail servers, and front-end relays.
6. Choosing a Messaging Server upgrade strategy for each host

   Three Messaging Server upgrade strategies offer choices that strike a balance between system downtime, cost, simplicity, and risk. You choose a strategy for each host, and you can use different strategies on different hosts within a Messaging Server deployment.
7. Upgrading the Messaging Server software

   Use Messaging Server 8.0.2 or the current patch.
8. (Optional) Migrating to Unified Configuration

   Use the **configtoxml** command to migrate from legacy configuration to Unified Configuration.

   For more information, see the discussion on the **configtoxml** command in *Messaging Server System Administrator's Guide*.

Technical Features Supporting Messaging Server Upgrade

The following features support Messaging Server upgrade:

- You migrate mailboxes by using the **imsbackup** and **imsrestore** commands. See the discussion on migrating mailboxes to a new system in the *Messaging Server System Administrator's Guide*. These commands support moving mailboxes from old message store versions to new ones (including when the message store database format changes, for example, from Messaging Server 32-bit to Messaging Server 64-bit). These commands also support moving mailboxes from new message store versions to old ones for back-out purposes.

  Note: You cannot upgrade a classic message store machine (imapd/popd/stored) directly to 8.0.2/Cassandra message store. You must use the **rehostuser** command to migrate mailboxes from Messaging Server 7.x, 8.0, and 8.0.1 to Messaging Server 8.0.2 and Cassandra message store. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

- In-place Upgrade supports changing the old mailbox format to the new format, but it does not support going from the new format back to the old. You **cannot** back out from new data format to old data format by using the in-place Upgrade Strategy. The conversion is done "on-the-fly" as mailboxes are accessed. In-place server upgrade is by done using **commpkg upgrade**.

- Alternate root (ALTROOT) install is supported on Oracle Solaris. For more information, see the discussion on using the ALTROOT command-line argument in "commpkg Reference".

Note: In general, you should avoid using the alternate root method when upgrading Messaging Server, because Solaris now uses alternate root for its Live Upgrade feature.

Messaging Server Upgrade Strategies

Messaging Servers supports the following three upgrade strategies for individual hosts. These strategies provide a balance between downtime, risk of extended downtime, complexity, and potential hardware costs.

- **Coexistent Upgrade:** You keep existing services online while you construct a new host on separate hardware.
- **Side-by-side Upgrade on the same host:** The new software version is installed on the same host as the old version in a different directory. After you migrate the software configuration to the new version, you switch the deployment over to the new version.
- **In-place Upgrade:** The binaries of the old version are replaced with the binaries of the new version on the same host. That is, you use **commpkg upgrade**.

The strategy chosen for any particular host might differ. For example, you might want to use an in-place upgrade on your front-end servers (relays, MMPs, and webmail servers) but you might want to do a coexistent upgrade on your message stores.

Caution: There is a data format change in the message store in Messaging Server 8.0.1 (see the discussion on upgrading the Message Store in the *Messaging Server System Administrator's Guide*). Coexistent upgrade is recommended to facilitate backing out from an upgrade. See also "Downgrading from Messaging Server 8.0.2" for additional information.

The strategy you chose also depends upon the version you currently have installed and whether you are using 32-bit or 64-bit Messaging Server product. Issues and compatibilities are described next.

Note: When upgrading/migrating between SPARC and x86 hardware, you need to use the Online/Coexistence strategy. For more information, see the discussion on migrating from x86 to SPARC in *Messaging Server System Administrator's Guide*.

Coexistent upgrade is also required when upgrading classic message store machines (imapd/popd/stored for 7.x, 8.0, and 8.0.1) to Cassandra message store.

The Coexistence Migration Strategy is the safest and most secure method of upgrading. It also has the lowest downtime of the three upgrade strategies. In the coexistence model, existing services remain online while you construct a new target host (or entire Messaging Server environment) on new hardware or in a Oracle Solaris whole root zone on the existing hardware. After the new host and environment are established, you can migrate a small number of friendly users to the new system to verify operations and administrative procedures. For a certain period both systems are accessible to user traffic. This is called a coexistence phase. Messaging access is not disrupted and proceeds invisibly to users. When all users are migrated to the new environment, you can decommission your legacy deployment. This phased approach ensures that the new system is fully prepared to handle production users before making the full migration.

Using the Coexistent Upgrade on Messaging Server

In this model of upgrading Messaging Server, you construct a new target host on a new hardware or in an Oracle Solaris whole root zone on the existing software. After the new host and environment are established, you can migrate users to the new system and decommission your legacy deployment.

Advantages and Disadvantages of Coexistence Migration

- Service downtimes are usually rare and short. There is less danger that they will be longer than the off-line windows imposed by service level agreements.
- Allows a gradual adoption of the new software so that you can gain confidence by trying it out with a small group of sympathetic users before migrating production users.
- The risk of upgrade failure is mitigated by the fact that your legacy system remains fully functioning throughout the upgrade process.
- Because the new system is built alongside a functional old one, you do not need to install or modify anything on the working legacy machines. This is an advantage as there is always a natural reluctance to modify or reconfigure a working legacy system in significant ways.
- Coexistence is the safest upgrade model and has the least amount of user downtime.
- Simpler back off procedure. Anytime you upgrade software, you need to make provisions for backing off from the new system to the old system in case of failure. Other upgrade models might require that you back up and turn off the old system, install, configure, and migrate to the new system. Only when you switch on the new system do you know if the upgrade succeeded. If it turns out, that it did not, then you might have to use your back off plan to put everything back into place. A coexistence migration is much simpler as a working legacy system is already in place.
- You must move user data, such as mailboxes, from one host to another. On classis message store, you typically use the **imsbackup** and **imsrestore** commands. To migrate to Cassandra message store, you must use the **rehostuser** command.
- Might require extra hardware to set up a parallel system. (This can be mitigated by upgrading legacy machines after they are no longer used.)

Specific Steps for Upgrading Messaging Server Using the Coexistence Model

The steps to upgrade Messaging Server using the Coexistence Model are as follows:

1. Make sure that your hardware is installed as per your Messaging Server deployment plan. For more information, refer to the previous chapters, or to *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for Cassandra requirements.

2. Install a new version of Messaging Server in the proper sequence on a new machine, by using the **commpkg install** command.
3. Configure Messaging Server.
   - You must do so manually. Basically, you must clone the relevant parts of the old machine's configuration to this new machine.
4. If you are doing a coexistent migration on a message store, migrate user mailboxes (a few at a time) to the new machine. For classic message store use the **imsbackup** and **imsrestore** commands. For Cassandra message store, use the **rehostuser** command. See *Messaging Server System Administrator's Guide* for more information.
   - Details on classic message store internals can be found in the *Messaging Server System Administrator's Guide.*

Note: To migrate mailboxes from classic message store to Cassandra message store, you must perform a number of steps, including installing message access layer hosts, configuring store affinity groups, installing Cassandra database, rehosting users, and so on. For more information, see *Messaging Server Installation and Configuration Guide for Cassandra Message Store*.

Using the Side-by-Side Upgrade on Messaging Server

Note: You cannot use the side-by-side strategy to upgrade a classic message store to Cassandra message store. Instead, you must perform a migration. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information.

In this model of upgrading Messaging Server, you install the new software on the same machine as the old version. The basic steps are as follows:

1. Back up configuration data (simply back up the configuration directory).

   For classic message store mailbox data, use the **imsbackup** command.
2. Install Messaging Server 8.0.2 side-by-side on the same machine with your earlier version of Messaging Server by using the **commpkg install** command.
3. Create a symbolic link for a level of indirection that you will use to point to the active Messaging Server installation.
4. Stop the currently running Messaging Server.
5. Point to the symbolic link to the Messaging Server 8.0.2 installation.
6. Start Messaging Server 8.0.2.

Advantages and Disadvantages of Side-by-Side Messaging Server Migration

- Second best minimal downtime.
- Second best in backout.
- Does not require extra machines.
- Does require different directory location for fresh install. Any custom scripts that reference the install location must be modified.

- Does not involve moving the classic store mailboxes. New version just “points” to the mailboxes and mailbox conversion to the new version is automatic and transparent.
- Backout is complicated and time consuming. See "Downgrading from Messaging Server 8.0.2."
- The only advantage of side-by-side over in-place is that the binaries of the old version remain intact on the system so you do not have to reinstall and reconfigure in the case of a backout.

Messaging Server 8.0.2 Side-by-Side Upgrade

Note: You cannot use the side-by-side strategy to upgrade a classic message store to Cassandra message store. Instead, you must perform a migration. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information.

This example describes how to upgrade from Messaging Server 7.0.5.31.0 to Messaging Server 8.0.2 by using the side-by-side method.

This section includes:

- Side-by-Side Migration Overview
- Side-by-Side Migration Example
- Handling Subsequent Upgrades

Side-by-Side Migration Overview

This example describes how to install both Messaging Server versions on the same host in separate directories, create a symbolic link to the active installation, then point the symbolic link at the single configuration and data location.

Note: Upgrading to Messaging Server 8.0.2 in a side-by-side installation works on both Solaris and Oracle Linux. This is not an alternate root installation as described in the discussion on ALTROOT command-line argument in "commpkg Reference". Due to package version changes starting with Messaging Server 8.0.1, you can use the method described in this information rather than the alternate root method, to simplify the upgrade process.

This example uses the following directories:

- **/opt/sun/comms/messaging64**: Directory in which Messaging Server 7.0.5.31.0 is installed (default location)
- **/var/opt/sun/comms/messaging64**: Directory containing the Messaging Server 7.0.5.31.0 data and configuration (default location)
- **/opt/ucs1/messaging64**: Directory in which Messaging Server 8.0.2 is installed (non-default location)

Additionally, this example uses the following symbolic link:

- **/opt/ucs/msg**: Symbolic link to either **/opt/sun/comms/messaging64** or **/opt/ucs1/messaging64**

Side-by-Side Migration Example

This section includes:

- Backing Up Messaging Server
- Creating the Symbolic Link for the Active Message Server Installation
- Installing and Configuring Messaging Server 8.0.2
- Changing Over from Messaging Server 7.0.5.31.0 to Messaging Server 8.0.2
- Post Upgrade

Backing Up Messaging Server

Before performing the upgrade, back up the system. See the following documentation for more information:

- See the discussion on best practices for Messaging Server and ZFS in *Messaging Server System Administrator's Guide*
- Downgrading from Messaging Server 8.0.2
- See the discussion on backing up and restoring the classic message store in *Messaging Server System Administrator's Guide*

Creating the Symbolic Link for the Active Message Server Installation

This example assumes that you have already installed and configured Messaging Server 7.0.5.31.0 in the default directory (**/opt/sun/comms/messaging64**), and that the Messaging Server is currently running.

1. Create a symbolic link for a level of indirection that you will use to point to the active Messaging Server installation.

   ```
   mkdir -p /opt/ucs
   cd /opt/ucs
   ln -s /opt/sun/comms/messaging64 msg
   ```

2. Ensure that external programs or plugins that refer to the Messaging Server installation use this symbolic link. Also, if you use Solaris Management Facility (SMF), ensure that you configure XML settings that start and stop Messaging Server to use this symbolic link.

Installing and Configuring Messaging Server 8.0.2

1. Change to the directory in which you have extracted the Messaging Server 8.0.2 media pack ZIP file.
2. Install Messaging Server 8.0.2 into its own directory, **/opt/ucs1**, by using the following **commpkg install** command.

   ```
   commpkg install --comp=MS64 --installroot /opt/ucs1 --silent=NONE
   ```

3. Configure Messaging Server 8.0.2 to point to the existing (Messaging Server 7.0.5.31.0) data and configuration location.

   ```
   cd /opt/ucs1/messaging64
   bin/useconfig /var/opt/sun/comms/messaging64/config
   ```

Changing Over from Messaging Server 7.0.5.31.0 to Messaging Server 8.0.2

1. Stop the currently running Messaging Server 7.0.5.31.0 processes.

   ```
   /opt/ucs/msg/bin/stop-msg
   ```

Note that this command actually uses the symbolic link to **/opt/sun/comms/messaging64**.

2. Change the symbolic link created previously to point to the Messaging Server 8.0.2 installation.

   ```
   cd /opt/ucs
   mv msg msg-old
   ln -s /opt/ucs1/messaging64 msg
   ```

3. Start the Messaging Server 8.0.2 processes.

   ```
   /opt/ucs/msg/bin/start-msg
   ```

   Note that this command actually uses the symbolic link to **/opt/ucs1/messaging64**.

Your deployment is now upgraded to Messaging Server 8.0.2

Post Upgrade

After completing the upgrade, remove the symbolic links (data, config, and log) in the previous Messaging Server installation. This is not a requirement, but a recommendation to protect against inadvertently using them.

```
cd /opt/sun/comms/messaging64
rm data config log
```

Handling Subsequent Upgrades

On the next upgrade, now that the two locations are populated, you can simply upgrade the inactive location. Following the preceding example, Messaging Server 8.0.2, installed in **/opt/ucs1** is active, and Messaging Server 7.0.5.31.0, installed in **/opt/sun/comms** is inactive.

1. Change to the directory in which you have extracted the latest Messaging Server version media pack ZIP file.
2. If you are upgrading from a Messaging Server version prior to 8.0.2, for example, 7.0.5.31.0, you must remove the symbolic links to the configuration and data, otherwise the uninstall stops the messaging services.

   ```
   cd /opt/sun/comms/messaging64
   rm config data log
   ```

3. Upgrade the inactive Messaging Server installation.

   ```
   commpkg upgrade --comp=MS64
   ```

   The upgrade prompts you to select the version that you want to upgrade. Specify the inactive version.

4. Change the symbolic link created previously to point to the new Messaging Server installation.

   ```
   cd /opt/sun/comms/messaging64
   bin/useconfig
   /var/opt/sun/comms/messaging64/config
   ```

5. Stop the running Messaging Server processes.

   ```
   /opt/ucs/msg/bin/stop-msg
   ```

   Note that this command actually uses the symbolic link to **/opt/ucs1/messaging64**.

6. Change the symbolic link created previously to point to the new Messaging Server 8.0.2 installation.

   Depending on which installation you are upgrading, use one of the following **ln** commands.

   ```
   cd /opt/ucs
   rm msg
   ln -s /opt/sun/comms/messaging64 msg
   <or, depending on which installation is upgraded>
   ln -s /opt/ucs1/messaging64 msg
   ```

7. Start the Messaging services using the new, upgraded version.

   ```
   /opt/ucs/msg/bin/start-msg
   ```

8. You should remove the symbolic links in the inactive installation, otherwise you might inadvertently use the inactive installation.

Using the In-Place Upgrade on Messaging Server

Note: You cannot use the in-place strategy to upgrade a classic message store to Cassandra message store. Instead, you must perform a migration. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information.

In this method you simply replace the old server binaries with the new server binaries on the same machine by using the **commpkg upgrade** command. This command removes the old packages and installs the new ones. For more information about this command, see the discussion on the **commpkg upgrade** command in "commpkg Reference".

Advantages and Disadvantages of In-place Messaging Server Upgrade

- Simplest. One command installs the old packages and removes the new packages. This command migrates and upgrades configuration.
- Requires least amount of extra disk space.
- Messaging Server stays in the same disk location (no tweaking of custom scripts).
- Has the most downtime.
- Back out is complicated and time consuming. See "Downgrading from Messaging Server 8.0.2."
- This method is probably best for evaluators/testers/developers.
- Useful for upgrading Messaging Servers configured without the message store, for example, front-end relays and webmail servers.

Specific Steps for Using In-Place Upgrade on Messaging Server

The following steps show how to upgrade Messaging Server using In-place Upgrade:

1. Run **commpkg upgrade** and select Messaging Server. This command:
   - Stops the servers
   - Removes the old version of Messaging Server

- Installs the new version of Messaging Server
- Performs a migration of configuration and classic store mailbox data

For information about using the **commpkg upgrade** command, see "commpkg Reference".

Downgrading from Messaging Server 8.0.2

If you upgrade using a coexistence migration strategy, you do not need to downgrade or back out a patch because you always have the system with the previous version of Messaging Server still running. Simply uninstall or decommission the newly installed version of Messaging Server on the new system and continue using the previous version on the old system. However, if you upgrade using a side-by-side or an in-place upgrade strategy, then you need to read the following information.

You cannot just back out the upgrade by using **commpkg uninstall** and then **commpkg install** from the previous release to reinstall the previous version. Instead, you must back up your Messaging Server data, back out the upgrade, then restore the Messaging Server data. For more information on the **commpkg uninstall** command, see "commpkg Reference".

These downgrade instructions apply to both the in-place or side-by-side upgrade methods.

Before you Upgrade to Messaging Server 8.0.2

Read this section before upgrading to Messaging Server 8.0.2.

- You cannot directly upgrade a classic message store to Cassandra message store. Instead, you must perform a migration. See *Messaging Server Installation and Configuration Guide for Cassandra Message Store* for more information.
- You cannot simply back out the Messaging Server 8.0.1 upgrade to a previous version once it is applied.
- Because you cannot directly upgrade a classic message store to Cassandra message store, the previous releases' downgrade instructions no longer apply.
- Although the system does permit you to back out the upgrade (for example, by running **commpkg uninstall** and then **commpkg install** from the previous release to reinstall the previous version, afterwards Messaging Server services may not properly start. Additionally, the **stored** process may not start properly, and any mailbox accessed prior to backing out the upgrade may report that it is corrupted with an invalid format. Furthermore, even if you could manage to start Messaging Server services and manually fix the mailbox corruption, the mailbox owner flags (for example, seen and deleted flags) are all reset.
- Before upgrading to Messaging 8.0.2, make sure that you back up the Messaging Server data. If you do need to downgrade the classic message store after upgrading to Messaging Server 8.0.2, you need to restore the Messaging Server data to its state prior to upgrading.
- Before upgrading to Messaging Server 8.0.2, it is highly recommended that you test it on a non-production system prior to actual deployment to production systems.
- Backing out from Messaging Server 8.0.2 is considered an avenue of last resort. If you need to downgrade, you must follow the steps described later in this

information to return your system to a working state. (Backing out is not relevant to a coexistence migration.)

- You need the previous version's software. For example, if you use the installer to upgrade from Messaging Server 7 Update 5, the installer removes the old software, and so to revert to that version, you would need the old product's installer to do so. Note that for scenarios other than classic store to Cassandra store migration, if you do a backup prior to downgrading, and restore from that backup, you do not lose messages that arrived since that backup when you restore.

Downgrading Using a ZFS Snapshot (Solaris Only)

To back out the upgrade on a host configured without a classic message store such as an MTA host, an MMP host, or a Webmail host, run **commpkg uninstall** and then **commpkg install** from the previous release to reinstall the previous version. On a host configured with a classic message store that uses a ZFS file system, you can use the following procedure to back out the upgrade without having to do a full **imsbackup/imsrestore** thereby taking advantage of the near instantaneous ZFS snapshot and roll back capability.

High Level Overview

Create a ZFS snapshot of the message store data including the **mboxlist** database, index, and message partitions **before** upgrading.

Once you upgrade, you can back out by:

- Performing an incremental **imsbackup** of the message store since the snapshot time
- Using **commpkg uninstall** and then **commpkg install** from the previous release to reinstall the previous version
- Rolling back to the ZFS snapshot
- Restoring the incremental **imsbackup**

Note however, that if you are backing out to a version prior to Messaging Server 7.0.5.29.0, those versions do not restore message flags from the incremental backup.

To Downgrade Using a ZFS Snapshot

1. Prior to upgrading, stop the services and create a ZFS snapshot of the classic message store. Note that in a subsequent step where a ZFS rollback is done to restore this snapshot, only the store area should be restored. In particular, you should not rollback the MTA queues. For additional information, see the discussion of ZFS best practices in *Messaging Server System Administrator's Guide*. Note the timestamp when you create the ZFS snapshot. We recommend using the timestamp in the name of the snapshot. The following example assumes that the store area is in the **rpool/export/comms-data** ZFS partition.

   ```
   stop-msg
   zfs list
   zfs snapshot rpool/export/comms-data@20150601232600
   ```

2. Upgrade and start services.

   ```
   commpkg upgrade
   start-msg
   ```

If you decide for whatever reason to downgrade, note that this decision should not be taken lightly. This should only be done if there is no other recourse.

3. Stop services.

```
stop-msg
```

4. Start Message Store services.

```
start-msg store
```

5. Do an incremental **imsbackup** from the time the ZFS snapshot was taken in Step 1 (timestamp 2015-Jun-01 11:26 pm).

```
imsbackup -x -v -f - -d 20150601:232600 / > /var/tmp/backup
```

> **Note:** It is better if the incremental backup is relatively small.

6. Stop services.

```
stop-msg
```

   It would seem prudent to do another ZFS snapshot prior to starting the downgrade, but ZFS snapshots do not allow you to rollback to more than the previous snapshot.

7. Uninstall the Messaging Server.

```
commpkg uninstall
```

8. Reinstall the previous Messaging Server version by running its installer.

```
commpkg install
```

9. Roll back to the ZFS snapshot you created previously.

```
zfs rollback rpool/export/comms-data@20150601232600
```

10. Start the message store services.

```
start-msg store
```

11. Restore the backup you made previously using **imsbackup** by running i**msrestore** with the -E switch.

```
imsrestore -E -f /var/tmp/backup
```

12. Start services.

```
start-msg
```

Downgrading from Messaging Server 8.0.2 Without Using a ZFS Snapshot

Use this procedure if you have upgraded to Messaging Server 8.0.2 and need to return to previous version.

To Downgrade from Messaging Server 8.0.2:

1. Prior to downgrading, perform a full backup of the classic message store by using the **imsbackup** command.

   For example:

```
stop-msg
```

```
start-msg store
imsbackup -v -f - / > backup
```

2. Uninstall the Messaging Server.

   ```
   commpkg uninstall
   ```

3. Reinstall the previous Messaging Server version by running its installer.

   ```
   commpkg isntall
   ```

4. Move the **store** directory to a different location.

   For example:

   ```
   mv data/store data/store.old
   ```

5. Start the message store to perform the restore:

   ```
   start-msg store
   ```

6. Perform the restore:

   ```
   imsrestore -f backup
   ```

7. Start Messaging Server.

   For example:

   ```
   start-msg
   ```

Downgrading MTA, MMP, or Webmail Hosts

To back out the upgrade on a host configured without a classic message store, such as an MTA host, an MMP host, or a Webmail host, run **commpkg uninstall** and then **commpkg install** from the previous release to reinstall the previous version.

Messaging Server 8.0.1 Upgrade in an HA Environment

Upgrading Messaging Server in a highly-available (HA) environment consists of upgrading the Messaging Server software then upgrading the Messaging Server Oracle Solaris Cluster Agent.

This section includes the following topics:

- Upgrading to Messaging Server 8.0.2 in an HA Environment
- Upgrading to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA)

Upgrading to Messaging Server 8.0.2 in an HA Environment

Upgrade strategies, each of which require different procedures, include the follow:

- Coexistent upgrade: This is similar to a fresh HA installation. See "Configuring Messaging Server for High Availability" for more information.
- Side-by-side upgrade
- In-place HA upgrade

To Do a Side-by-side Upgrade to Messaging Server 8.0.2 in an HA Environment

1. Go to resource group online node.

a. Disable Messaging Server resource.

```
scswitch -n -j msg_svr_resource
```

b. Upgrade Messaging Server by using the side-by-side strategy. See "Using the Side-by-Side Upgrade on Messaging Server" for more information. Perform this step only on the Messaging Server resource group online node. Do not start Messaging Server yet.

c. Run the **ha_ip_config** command on the Messaging Server resource group online node.

```
MessagingServer_home/bin/ha_ip_config
```

This command is needed only if the currently installed Messaging Server is prior to version 7.0.

d. Start the watcher process once on the Messaging Server resource group online node.

```
start-msg watcher
```

2. Switch over to other node:

```
scswitch -z -g msg_svr_resource_group -h node-name
```

3. Run the **useconfig** command.

   This is needed if you are upgrading Messaging Server from 32-bit to 64-bit, to update the trusted library path for 64-bit applications to include Messaging Server **/bin/crle -s -64** *new_MessagingServer_home***/lib')**.

```
MessagingServer_home/bin/useconfig MessagingServer_home/config
```

4. Change **IMS_serverroot** path for Messaging Server resource if new Messaging Server base directory is different from old installation.

```
scrgadm -cj msg_svr_resource -x IMS_serverroot=new_MessagingServer_home
```

5. If Messaging Server Oracle Solaris Cluster agent (**MS_SCHA**) is old (not from Communications Suite 6 or later), then it does not work with upgraded Messaging Server and you need to perform the **MS_SCHA** upgrade procedure.
6. Enable Messaging Server resource.

```
scswitch -e -j msg_svr_resource
```

To Do an In-place Upgrade to Messaging Server 8.0.2 in an HA Environment

An in-place upgrade is done by using the **commpkg upgrade** command.

1. Disable Messaging Server resource:

```
scswitch -n -j msg_svr_resource
```

2. Run the **commpkg upgrade** command on all nodes of the cluster
3. Run the **ha_ip_config** command on the Messaging Server resource group online node.

```
MessagingServer_home/bin/ha_ip_config
```

   This command is needed only if the currently installed Messaging Server is prior to version 7.0.

4. Start the watcher process once on the Messaging Server resource group online node.

   ```
   start-msg watcher
   ```

5. Enable Messaging Server resource:

   ```
   scswitch -e -j msg_svr_resource
   ```

Upgrading to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA)

This section provides instructions for the Oracle Solaris Cluster Agent upgrade. It consists of the following sections:

- To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA)
- To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA) if Cluster Nodes Include Non-Global Zones
- To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA) in a Two-node Symmetric Oracle Solaris Cluster HA Environment

To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA)

1. Run **commpkg upgrade** on all nodes on the cluster.

   Messaging Server should be upgraded to 8.0.q before upgrading Messaging Server Oracle Solaris Cluster Agent.

2. Enable Messaging Server resource:

   ```
   scswitch -e -j msg_svr_resource
   ```

To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA) if Cluster Nodes Include Non-Global Zones

If a machine that has non-global zones participates in a cluster, all zones on that machine must be in the cluster. The Oracle Solaris Cluster software and HA agents should be installed in all zones, and **MS_SCHA** should be installed in the global zone and automatically propagated into all non-global zones (that is, don't use the **-G** switch to **pkgadd**). The Messaging Server Installer treats HA agents like **MS_SCHA** as a product that should be propagated to all non-global zones when it is installed in the global zone. In the rare case where you have managed to install the pre-version 7 **MS_SCHA** agent in the non-global zones, then an upgrade consists of first uninstalling the older agent from all non-global zones, followed by installing the new 7 **MS_SCHA** agent in the global zone.

To check if the older pre-version 7 agent was installed in the global zone and automatically propagated to all non-global zones, verify that **SUNWscims** is listed in **/var/sadm/install/gz-only-packages**. If it is, then run commpkg upgrade in the global zone. If it is not listed, then **SUNWscims** is either not installed, or is installed so that it is propagated to non-global zones. If this is this case, use the following procedure:

1. Run **commpkg uninstall** and uninstall **MS_SCHA** in every non-global zone (do not uninstall it in the global zone).
2. In the global zone, run **commpkg upgrade** and upgrade **MS_SCHA**.

To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA) in a Two-node Symmetric Oracle Solaris Cluster HA Environment

1. Upgrade Messaging Server to Version 8.0.2 before upgrading the Messaging Server Oracle Solaris Cluster Agent.
2. Make sure that the Messaging Server installation location is accessible from both nodes.

   This is required because a resource type upgrade command validates accessibility. For the first instance in a Symmetric Cluster setup, Messaging Server installation is done on first node only (on a shared storage mount point). For the second instance, Messaging Server installation is done on second node only.
3. Follow the steps mentioned in "To Upgrade to the Messaging Server Oracle Solaris Cluster Agent (MS_SCHA)."

Note: If you prefer to upgrade Oracle Solaris Cluster Agent (**MS_SCHA**) for only one instance, then follow the prior steps and correct the resource type version using Oracle Solaris Cluster commands.

Messaging Server Upgrade in Silent Mode

When you run the installer to upgrade in silent mode, you are running a non-interactive session. The upgrade 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 upgrade multiple instances of the same software and configuration without having to manually run an interactive upgrade for each instance.

To Run a Messaging Server Silent Upgrade

To run a Messaging Server silent upgrade:

1. Obtain the state file by one of the following two means:
   - Run an interactive upgrade 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 upgrade --dry-run
     ```
2. Copy the state file to each host and edit the file as needed. See Silent Mode File Format.
3. Run the silent installation on each host.

   For example:

   ```
   commpkg upgrade --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 the discussion on the silent installation usage in "commpkg Reference".

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

Note: If you specify *None* for the silent file, then defaults are used for the property values.

Silent Mode File Format

The silent mode file (also known as a state file) is formatted like a property file: comment lines begin with a number sign (**#**) and properties are key/value pairs separated by an equals (**=**) sign. Table 13–1 displays the changes you can make to the following options. For more information on the **commpkg upgrade** options, see the discussion on **commpkg upgrade** options in "commpkg Reference".

Table 13–1 Silent Mode File Options

Option	Description	Example
VERB	Indicates which function to perform. You can add CLI arguments described in "commpkg Reference," however the **---dry-run argument** cannot be added to the **install** function in the state file.	VERB=upgrade
USEPKGUPGRADE	Indicate whether to perform upgrade by using **pkgrm** and **pkgadd** commands.	USEPKGUPGRADE=no
UPGRADESC	Indicates whether all shared components should or should not be upgraded without prompting.	UPGRADESC=no
PKGOVERWRITE	Forces the overwriting of the existing installation packages even if patches are available to do the upgrade.	PKGOVERWRITE=YES
INSTALLROOT	Specifies installation root.	INSTALLROOT=/opt/sun/comms
EXCLUDESC	Specifies to exclude shared component patches.	EXCLUDESC=no
EXCLUDEOS	Specifies to not upgrade operating system patches.	EXCLUDEOS=YES
COMPONENT_ VERSIONS	unused	COMPONENT_VERSIONS= 7.4 6.4 7.2 8.3 2.0 1.3 7.0
COMPONENTS	Lists the components you want to upgrade.	COMPONENTS=MS64 to specify 64-bit Messaging Server
ALTROOT	Specifies an alternate root.	ALTROOT=yes
ALTDISTROPATH	Indicates an alternate distro path if --distro is not specified.	ALTDISTROPATH=SunOS5.10_i86pc_ DBG.OBJ/release

To Display Product Mnemonic Names

To display a complete list of the mnemonic product names (such as MS, MS64, CS) to use with the **COMPONENTS** property, run the **commpkg info --listPackages** command.

14 Uninstalling Messaging Server

This chapter describes how to uninstall Oracle Communications Messaging Server.

About Uninstalling Messaging Server

The following steps gives a high-level overview on how to uninstall Messaging Server.

To uninstall Messaging Server:

1. Log in as **root**.
2. Change to the *InstallRoot* directory.
3. Run the **commpkg uninstall** command.
4. Choose Messaging Server.
5. When prompted, enter **Yes** to continue.

Uninstalling Messaging Server

The **commpkg uninstall** command enables you to uninstall Messaging Server products and shared components.

For information, see the discussion on the **commpkg** general syntax, other commands and options, in "commpkg Reference".

Uninstalling Messaging Server Components

This command uninstalls Messaging Server products. However it does not remove OS patches or shared components installed by **commpkg install**.

To uninstall one or more Messaging Server component, change to the *InstallRoot* directory, and as **root**, run **commpkg uninstall**

Note: A fast way to uninstall a Messaging Server component installed in an alternate root is to simply remove the entire alternate root directory.

commpkg uninstall Command Syntax

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

You must be logged in as superuser **(root)** to run this command.

Using the ALTROOT | name Command-line Argument

This argument is supported only on Solaris OS. Specify **ALTROOT | name** on the command line to uninstall an alternate root directory for the uninstallation.

If you specify the **--rootdir** option in addition to the **ALTROOT | name** command-line argument, they must match.

For more information, see the discussion on using the ALTROOT command-line argument in "commpkg Reference".

commpkg uninstall Command Options

The following options are used by the **commpkg uninstall** command:

Table 14–1 commpkg uninstall Command Options

Options	Description
--silent *INPUTFILE*	Runs the uninstaller silently, taking the inputs from the *INPUTFILE* and the command-line arguments. Any command-line arguments override entries in the *INPUTFILE*. Uninstallation proceeds without interactive prompts. Use **--dry-run** to test silent uninstallation.
--dry-run or **-n**	Does not actually uninstall Messaging Server components, only performs checks. The silent uninstallation *INPUTFILE* is created in **/tmp**.
--rootdir *path*	This option is deprecated in favor of using the **ALTROOT** or **name** command-line argument. This option specifies the path of *ALTROOT*, the alternate root used for multi-installation. Supported on Solaris OS only. If you specify this option and the *ALTROOT | name* argument the values must be consistent.

Uninstalling Messaging Server in Silent Mode

If you run the uninstaller in Silent mode, you are running a non-interactive session. The uninstallation inputs are taken from a silent uninstallation file (also known as a state file), from command line arguments, or defaults.

To run a silent uninstallation, follow these steps:

1. Run an interactive uninstallation session.

   A state file similar to **/var/opt/CommsInstaller/logs/silent_CommsInstaller_20130130090040** is automatically created.

   Note: The silent installation and uninstallation files have the same syntax and file naming convention. To determine if it is an installation or uninstallation silent file, consult the value of the VERB property.

2. Copy the state file to each host machine and edit the file as needed.
3. Run the silent uninstallation on each host.

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

15 Installing Patches

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

See the patch ReadMe file, included in the patch download, for information about the contents of a patch.

About Patching Messaging Server

Messaging 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 Messaging 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 Messaging Server and verifying the version number of installed components.

Installing a Patch

To install a patch on Messaging Server, run the following command:

```
commpkg upgrade
```

For more information, see "upgrade Verb Syntax".

A Messaging Server Configuration Scripts

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

configure Script

The **configure** script enables you to perform an initial configuration of your Messaging Server deployment. Table A–1 describes the **configure** options.

Table A–1 configure Options

Option	Description
--dataroot=*path*	Specifies a non-default *DataRoot*. The product finds a non-default *DataRoot* by following the *InstallRoot***/data** symlink, and finds a non-default *ConfigRoot* by following the *InstallRoot***/config** symlink. Starting with Messaging Server 8.0.2, those two symlinks are optional and need not be present.
--debug	Provides general debug information primarily for LDAP operations.
--help	Displays help.
--ignoreSendmail	Keeps **sendmail** enabled after configuration. In other words, **sendmail** does not disable after configuration.
--ldapport [*ldapport*]	Specifies an LDAP port other than the default port 389.
--ldif	Causes configure to run without modifying the directory and instead generate an LDIF file (*MessagingServer_home***/data/install/configure.ldif**) which the administrator can apply to the directory after initial configuration. This is needed if the person doing the installation does not have directory administrative rights.
--noldap	Runs without LDAP present (statefile only).
--novalidate	Skips most validation of user input.
--noxml	Generates legacy configuration (does not use XML-based Unified Configuration); can also be used to replace a Unified Configuration with a freshly generated legacy configuration (fresh installation of Messaging Server, not an upgrade where the **configtoxml** command was run). Not supported as of Messaging Server 8.0.2.
--preserveCritical	Does not overwrite critical attributes.
--saveState [*statefile*]	Specifies a location other than the default location (mentioned below) to save a state file.
--ssl	Requires SSL when configuring LDAP.
--state [*statefile*]	Uses a silent installation file. See "To Run a Messaging Server Silent Installation".

Table A–1 (Cont.) configure Options

Option	Description
--version, **--V**	Displays product version.
--xml	Generates Unified Configuration (XML).
--noisc	Configures the front-end MTA, MMP, and **mshttpd** hosts to not use the Indexed Search Converter (ISC). With the removal of the classic message store in this release of Messaging Server, this enables you to install front-end services without requiring that Java be present.

Table A–2 describes the options that you can use in the **configure** statefile.

Table A–2 configure statefile Options

Option	Description
Fqdn.TextField	Specifies local host's fully qualified domain name.
msg.DataPath	Specifies the non-default *DataRoot*.
iMS.UserId	Specifies the user name for server processes.
iMS.GroupId	Specifies the group name for server processes. Only used if **iMS.UserId** doesn't exist and needs to be created, in which case this group is used.
UGDIR_URL	Specifies the ldap or ldaps URL for the Directory Server.
UGDIR_BINDDN	Specifies the LDAP administrator user.
UGDIR_BINDPW	Specifies the LDAP administrator password. This password is obfuscated using a ROT-13 variant.
Postmaster.TextField	Specifies the email address for the postmaster.
admin.password	Specifies the password for server administration. This password is base64-encoded for obfuscation.
EmailDomain.TextField	Specifies the default email domain.
OrgName.TextField	Specifies the existing default organization DN.
InternalIPlist	Specifies IP address information, in comma-delimited form, for systems permitted to relay mail without authentication.
XMLCONFIG	When set to 1, creates a Unified Configuration. When set to 0, creates a legacy configuration. The **--xml** and **--noxml** command options override what is specified in the statefile. For Messaging Server 8.0.2, this option is ignored and Unified Configuration is always used.
cassandra	When set to 1, creates a Cassandra message store.
dssetup.ugsuffix	Specifies the user/group suffix.
dssetup.dcsuffix	Specifies the domain suffix.
dssetup.schematype	Specifies the schema type.

B

commpkg Reference

This appendix provides information about the **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 Oracle Communications Messaging Server software and its shared components. The **commpkg** command is installed in the directory in which you unzip the product software.

Syntax

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

Table B–1 describes the **commpkg** general options.

Table B–1 commpkg 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 B–2 describes the installer verbs.

Table B–2 Installer Verbs

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

Table B–2 (Cont.) Installer Verbs

Verb	Description
info	Displays product information on the paths (also known as *installroots*) where Messaging 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 B–3 displays the options for the **commpkg install** command.

Table B–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** *path*	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 B–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* if you want 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 needs to 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 need to 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 [uninstall_options] [ALTROOT|name]
```

Table B–4 displays the options for the **commpkg uninstall** command.

Table B–4 commpkg uninstall Options

commpkg install Options	**Description**
-? or **--help**	Displays help.
-V or **--version**	Displays the Installer version.

Table B–4 (Cont.) commpkg uninstall Options

commpkg install Options	Description
--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 [upgrade_options] [ALTROOT|name]
```

Table B–5 displays the options for the **commpkg upgrade** command.

Table B–5 commpkg upgrade Options

Options	Description
-? or **--help**	Displays help.
-V or **--version**	Displays the Installer version.
--excludeOS	Does not apply operating system patches during product upgrade.
--excludeSC	Does not install, upgrade, or patch any shared components.
ALTROOT | *name*	This option is available on Solaris only. Specifies an alternate root directory during a multiple host installation. The *InstallRoot* (the top-level installation directory for all products and shared components) is the alternate root. If you specify a *name*, it is an "alias" associated with the alternate root that is registered in the software list. You can use this option to upgrade multiple product instances on the same host or Solaris zone. Additionally, you can use this option to perform a side-by-side product upgrade.
--distro *path*	Specifies the *path* to packages and patches for the products. Default path: Location of the **commpkg** command.

Table B–5 (Cont.) commpkg upgrade Options

Options	Description
--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* if you want 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 or not to upgrade shared components as required. If this option is not specified, you are prompted for each shared component that needs to 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 [verify_options] [ALTROOT|name]
```

Tip: When verifying the package installation in an alternate root, be aware that Messaging 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 B–6 displays the options for the **commpkg verify** command.

Table B–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 need to 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* if you need to specify an alternate root directory on which to perform the package verification.

info Verb Syntax

```
commpkg info [info_options] [ALTROOT|name]
```

Table B–7 displays the options for the **commpkg info** command.

Table B–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 make up each Messaging Server, shared components, and operating system auxiliary product. This option also displays the mnemonic for Messaging Server and components such as **comm_dssetup.pl**.
--verbose	Prints product information installed in the *installroots*. To print information for a specific *installroot*, run the following command: **commpkg info --verbose** *installroot*
--components *comp1 comp2...*	A space delimited set of component products. Each product has mnemonic associated with it. Use **commpkg info --listPackages** to see the mnemonic for a product. In most shells you need to escape the space between each mnemonic, that is, by adding double quotes around all the components.

Using the Alternate Root Option

This section describes how to use the alternate root option to install multiple copies of the same product version on the same Solaris machine or Solaris zone.

About the Alternate Root

The Installer enables you to 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 Messaging 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 Messaging Server on the same physical host, another option is to use Solaris zones. For more information, see "Installing Messaging Server on Solaris Zones".

ALTROOT|name Syntax and Examples

The Installer uses the optional *ALTROOT* | *name* option to 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* in place of 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 Messaging 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 Messaging 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 Messaging 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 Messaging 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 need to take steps to resolve the port conflicts.

C

comm_dssetup.pl Reference

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

Before Running the comm_dssetup.pl Script

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

About 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.

   If you are not using Oracle Directory Server Enterprise Edition, or have customized your Directory Server, stop the process here without running the script. For more information, see "Directory Server Considerations for the comm_dssetup.pl Script".

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

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

Directory Server Considerations for the comm_dssetup.pl Script

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

- **comm_dssetup.pl** configures local Directory Server instances, and thus you must:
  - Install the **comm_dssetup.pl** script on every host on which a Directory Server instance resides.
  - Run the **comm_dssetup.pl** script on the same host as your Directory Server. The tool runs locally for a specific instance (specified by path of Directory Server or path of instance).

- You can run the **comm_dssetup.pl** script against any Directory Server instance on the local host. If you have multiple Directory Information Trees (DITs) on one host, you can maintain and update one installation of **comm_dssetup.pl**, and apply it to every Directory Server instance on the host.
- **comm_dssetup.pl** must configure every Directory Server instance for the same DIT. This assumes that:
  - A Directory Server has already been installed, configured, and is running before you launch the **comm_dssetup.pl** script.
  - When adding an additional Directory Server host (such as a replica), at a future date, you must run the **comm_dssetup.pl** script against it, too.
- If you have customized your Directory Server, the following considerations might apply:
  - If you have indexed some attributes, you might have to reindex those attributes after running the **comm_dssetup.pl** script.
  - If you have added other LDIF files (schema definitions), they should not be affected, so no action should be necessary. However, back up your custom schema definition files before running the **comm_dssetup.pl** script.

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

Information Needed to Run the comm_dssetup.pl Script

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

Table C–1 comm_dssetup.pl Information

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

Table C–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 (as well as the Directory Server binaries) resided. After Directory Server 6, the concept of the "server root" was removed. Directory Server instances (as well as 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

Messaging 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 Messaging Server for the first time and you do not have other applications in your deployment that are dependent on a two-tree provisioning model.

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

About LDAP Schema 1

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

About LDAP Schema 2 Compatibility Mode

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

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

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

Attribute Indexes Created by the comm_dssetup.pl Script

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

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

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

Table C–2 Attributes Indexed by comm_dssetup.pl

Suffix	Attributes Indexed	Type of Indexes Added
User/Group	**mail**	**pres, eq, approx, sub**
User/Group	**mailAlternateAddress**	**pres, eq, approx, sub**
User/Group	**mailEquivalentAddress**	**pres, eq, approx, sub**
User/Group	**mailUserStatus**	**pres, eq**
User/Group	**member**	**eq**
User/Group	**ou**	**pres**
User/Group	**cosspecifier**	**pres**
User/Group	**groupid**	**pres, eq, sub**
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 C–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 need to manually run the **dssetup.sh** script that was created.

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

Silent Mode Options

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

Table C–3 comm_dssetup.pl Silent Mode Options

Option and Argument	Description
-i yes | **no**	Specifies whether to configure new indexes. **yes** - Add new Directory Server indexes. **no** - Do not add indexes.
-R yes | **no**	Specifies whether to reindex automatically. **yes** - Reindex without prompting the user. **no** - Do not reindex without prompting the user. The **-m** option must also be specified for **yes** for the **-R** option to take effect.
-c *DirectoryServerRoot*	Specifies the Directory Server root path, for example, **/var/opt/sun/directory**.
-d *DirectoryInstance*	Specifies the Directory Server instance subdirectory under the Directory Server root path, for example, **slapd-varrius**.
-r *DCTreeSuffix*	Specifies the DC tree root suffix (for Schema 1 and Schema 2 compatibility modes only), for example, **o=internet**.
-u *UserGroupSuffix*	Specifies the user and group root suffix, for example, **o=usergroup**.
-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. This option is only used for Messaging Server 6.2 or prior.
-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**.