Sun Java™ System

Messaging Server 6 Deployment Planning Guide

2004Q2

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 817-6440-10

Contents

List of Figures

List of Tables

Preface

The *Sun Java System Messaging Server Deployment Planning Guide* contains the information you need to deploy Sun™ Java System Messaging Server 6 2004Q2 and its accompanying software components. Messaging Server provides a powerful and flexible cross-platform solution to meet the email needs of enterprises and messaging hosts of all sizes using open Internet standards.

This preface contains the following sections:

Who Should Read This Guide

You should read this book if you are responsible for deploying Messaging Server at your site.

NOTE You will not be able to directly migrate any existing mailboxes and message queues from Netscape Messaging Server or Sun Internet Mail Server products to Sun Java System Messaging Server.

Refer to the *Sun ONE Messaging Server 5.2 Migration Guide* to migrate from Netscape Messaging Server or Sun Internet Mail Server to Sun ONE Messaging Server 5.2. Then, follow the instructions in the *Sun Java System Messaging Server Administration Guide*, to upgrade from Messaging Server 5.2 to Sun Java System Messaging Server 6.

What You Need to Know

This book assumes that you are responsible for planning the Messaging Server deployment and that you have a general understanding of the following:

- The Internet and the World Wide Web
- IMAP, POP, SMTP, HTTP, and LDAP protocols
- Sun Java™ Enterprise System
- Sun Java™ System Administration Server
- Sun Java™ System Identity Server
- Sun Java™ System Web Server
- Sun Java™ System Directory Server
- Sun Java™ System Console
- Solaris™ system administration and networking

Conventions

The following table describes the typeface conventions used in this guide.

Table 1 Typeface Conventions

Typeface	Meaning	Examples
`AaBbCc123` (Monospace)	API and language elements, HTML tags, Web site URLs, command names, file names, directory path names, on-screen computer output, sample code.	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`
`AaBbCc123` (Monospace bold)	What you type, as contrasted with on-screen computer output.	`% su` `Password:`
AaBbCc123 (Italic)	Book titles.	Read Chapter 6 in the *User's Guide*.
	New words or terms.	These are called *class* options.
	Words to be emphasized.	You *must* be superuser to do this.
	Command-line variables to be replaced by real names or values.	The file is located in the *ms_svr_base*`/sbin` directory.

The following table describes placeholder conventions used in this guide.

Table 2 Placeholder Conventions

Item	Meaning	Examples
product_base	Placeholder for the directory where the product is installed.	The *ms_svr_base*`/bin` directory might be `/opt/SUNWmgsr`.

The following table describes the symbol conventions used in this book.

Table 3 Symbol Conventions

Symbol	Meaning	Notation	Example
`[ ]`	Contain optional command options.	`O[`*n*`]`	`O4, O`
`{ }`	Contain a set of choices for a required command option.	`d{y|n}`	`dy`
`|`	Separates command option choices.		

Table 3 Symbol Conventions *(Continued)*

Symbol	Meaning	Notation	Example
+	Joins simultaneous keystrokes in keyboard shortcuts that are used in a graphical user interface.		Ctrl+A
-	Joins consecutive keystrokes in keyboard shortcuts that are used in a graphical user interface.		Esc-S
>	Indicates menu selection in a graphical user interface.		File > New File > New > Templates

Resources on the Web

In addition to this guide, Messaging Server comes with supplementary information for administrators as well as documentation for end users and developers. Use the following URL to see all the Messaging Server documentation:

`http://docs.sun.com/coll/MessagingServer_04q2`

The Messaging Server product suite contains other component products, such as Sun Java System Console, Sun Java System Directory Server, and Sun Java System Administration Server. Documentation for these and other products can be found at the following URL:

`http://docs.sun.com/coll/entsys_04q2`

Third-party URLs are included in this document to provide additional, related information.

NOTE Sun is not responsible for the availability of third-party Web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Also, see the Sun Java System Messaging Server Software Forum for technical help on specific Messaging Server product questions. The forum can be found at the following URL:

http://swforum.sun.com/jive/forum.jsp?forum=15

How to Report Problems

If you have problems with Messaging Server, contact Sun customer support using one of the following mechanisms:

- Sun Software Support services online at

  http://www.sun.com/service/sunone/software

  This site has links to the Knowledge Base, Online Support Center, and ProductTracker, as well as to maintenance programs and support contact numbers.

- The telephone dispatch number associated with your maintenance contract

So that we can best assist you in resolving problems, please have the following information available when you contact support:

- Description of the problem, including the situation where the problem occurs and its impact on your operation
- Machine type, operating system version, and product version, including any patches and other software that might be affecting the problem
- Detailed steps on the methods you have used to reproduce the problem
- Any error logs or core dumps

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. Use the web-based form to provide feedback to Sun:

http://www.sun.com/hwdocs/feedback/

Please provide the full document title and part number in the appropriate fields. The part number is a seven-digit or nine-digit number that can be found on the title page of the book or at the top of the document. For example, the part number of this *Sun Java System Messaging Server Deployment Planning Guide* is 817-6440-10.

Chapter 1

Introduction to Messaging Server Software

Sun Java™ System Messaging Server 6 2004Q2 is a powerful, standards-based Internet messaging server designed for high-capacity, reliable handling of the messaging needs of both enterprises and service providers. The server consists of several modular, independently configurable components that provide support for several standards-based email protocols.

Messaging Server uses a centralized LDAP database for storing information about users, groups, and domains. Some information about server configuration is stored in the LDAP database. Some information is stored in a set of local configuration files.

The Messaging Server product suite provides tools to support user provisioning and server configuration.

This chapter contains the following sections:

Support for Standard Protocols

Messaging Server supports most national, international, and industry standards related to electronic messaging. For a complete list, see Appendix A of the *Sun Java System Messaging Server Administration Reference*:

http://docs.sun.com/doc/817-6267

Support for Hosted Domains

Messaging Server provides full support for hosted domains—email domains that are outsourced by an ISP. That is, the ISP provides email domain hosting for an organization by operating and maintaining the email services for that organization remotely. A hosted domain can share the same Messaging Server host with other hosted domains. In earlier LDAP-based email systems, a domain was supported by one or more email server hosts. With Messaging Server, many domains can be hosted on a single server. For each hosted domain, there is an LDAP entry that points to the user and group container for the domain and provides various domain-specific default settings.

Support for User Provisioning

Messaging Server uses a centralized LDAP database for storing information about users, groups, and domains. At this time, Messaging Server supports two schema options, Sun Java™ System Schema 1 (Schema 1) and Sun Java™ System Schema 2 (Schema 2). The provisioning options will depend on which schema you have chosen. See Chapter 11, “Pre-Installation Considerations and Procedures” for more information.

At this time, Messaging Server provisioning for Schema 2 can only be done using the `commadmin` utility, as documented in the *Sun Java System Communications Services User Management Utility Administration Guide*:

http://docs.sun.com/doc/817-5703

Schema 1 is supported by the iPlanet™ Delegated Administrator for Messaging product, which provides a graphical user interface and a set of command-line utilities for managing the users, groups, and domains within an organization. You can also use the following documentation, pertaining to previous software releases, for managing users, groups, and domains in Schema 1:

- *iPlanet Messaging Server Provisioning Guide* - Describes how to create domain, user, group, or administrator entries using LDAP:

  http://docs.sun.com/doc/816-6018-10

- *iPlanet Messaging and Collaboration Schema Reference Manual* - Describes Schema 1 for Messaging Server:

  http://docs.sun.com/doc/816-6021-10

- *iPlanet Messaging Server Reference Manual* - Describes the iPlanet Delegated Administrator command-line utilities for managing users, groups, and domains:

  http://docs.sun.com/doc/816-6020-10

- iPlanet Delegated Administrator online help

NOTE The Sun Java System Identity Server console provides minimal Messaging Server and Calendar Server LDAP user entry provisioning using Identity Server Services. Because the interface provides no input validation, user entries that cannot receive email or otherwise don't function will be created without reporting any errors. As a result, use this interface for demonstration purposes only.

The `commadmin` interface, which is described in the *Sun Java System Communications Services User Management Utility Administration Guide*, is the recommended mechanism for provisioning Messaging Server users.

Support for Unified Messaging

Messaging Server provides the basis for a complete unified messaging solution: the concept of using a single message store for email, voicemail, fax, and other forms of communication.

Support for Webmail

Messaging Server includes Messenger Express, a web-enabled electronic mail program that lets end users access their mailboxes using a browser running on an Internet-connected computer system using HTTP. Messenger Express clients send mail to a specialized web server that is part of Messaging Server. The HTTP service then sends the message to the local MTA or to a remote MTA for routing or delivery.

NOTE Sun Java™ System Communications Express also supports the Messenger Express client. See the Communications Express documentation for more information:

http://docs.sun.com/coll/MessagingServer_04q2

Powerful Security and Access Control

Messaging Server provides the following security and access control features:

- Support for password login and certificate-based login to POP, IMAP, HTTP, or SMTP
- Support for standard security protocols: Transport Layer Security (TLS), Secure Sockets Layer (SSL) and Simple Authentication and Security Layer (SASL)
- Delegated administration through access-control instructions (Schema 1 only)
- Client access filters to POP, IMAP, SMTP, and HTTP
- Filtering of unsolicited bulk email using system-wide, per-user, and server-side rules

Convenient User Interfaces

Messaging Server consists of several modular, independently configurable components that provide support for email transport and access protocols.

To configure the Message Transfer Agent (MTA), Messaging Server provides a complete set of command-line utilities and configuration files stored locally on the server. To configure the Message Store and message access services, Messaging Server provides a console graphical user interface and a complete set of command-line utilities.

Chapter 2

Analyzing Your Requirements

Planning your Messaging Server deployment requires that you first analyze your organization's business and technical requirements. This chapter helps you to gather and assess your requirements, which you then use to determine your Messaging Server architecture.

This chapter contains the following sections:

Identifying Deployment Goals

Before you purchase or deploy Messaging Server software or hardware, 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.

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

- Business requirements
- Technical requirements
- Financial requirements
- Service Level Agreements (SLAs)

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 accepted by the stake holders of the project is precarious at best.

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 affect 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.

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. The following questions will help you determine such goals.

- How are current services utilized?
- Can your users be categorized (for example, as sporadic, frequent, or heavy users)?
- What size messages do users commonly send?
- How many messages do users typically send per day or per hour?
- To which sites in your company do your users send messages?

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, 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

The following questions help you understand your network requirements:

- Do you want to obfuscate internal network information?
- 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?
- Do you want to reduce network HTTP traffic?

NOTE Answering yes to these questions suggests a two-tier architecture. See Chapter 3, "Developing a Messaging Architecture" for more information.

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?

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 associated with the project by applying sufficient attention and analysis to the many factors associated with the project requirements.

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.

Determining Project Goals

Your investigation and analysis should reveal your 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 projects 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 timeframe that you can plan for. Typically, a growth timeline is in the range of six to twelve months. 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, 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. 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. The following table presents some factors to consider as to whether to deploy more smaller hardware systems or fewer larger hardware systems.

Table 2-1 Considerations for Total Cost of Ownership

Hardware Choice	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 also mean a greater system down-time for maintenance, upgrade and migration.

Developing a Messaging Architecture

This chapter describes how to design the architecture of your Messaging Server deployment.

This chapter contains the following sections:

Purpose of a Messaging System Architecture

A good email system architecture quickly delivers email with embedded sound, graphics, video files, HTML forms, Java applets, and desktop applications, while providing for future upgrade and scalability. At a simplistic level, the Messaging Server architecture should:

- Accept mail from external sites
- Determine the user mailbox to deliver that message to and route it accordingly
- Accept mail from internal hosts
- Determine the destination system to deliver that message to and route it accordingly

Central to an email system architecture is the Messaging Server, a collection of components used to send and deliver messages. In addition to components provided in Messaging Server, the email system also requires an LDAP server and a DNS server. Many enterprises have an existing LDAP server and database that can be used with Messaging Server. If not, Java Enterprise System provides an LDAP server (Sun Java System Directory Server). The DNS server must be in place before deploying your email system.

The remainder of this chapter describes the components of the Messaging Server used to design an efficient scalable messaging system, as well as the Messaging Server software architecture.

Several factors other than efficiency and scalability influence the Messaging Server architecture. Specifically these are:

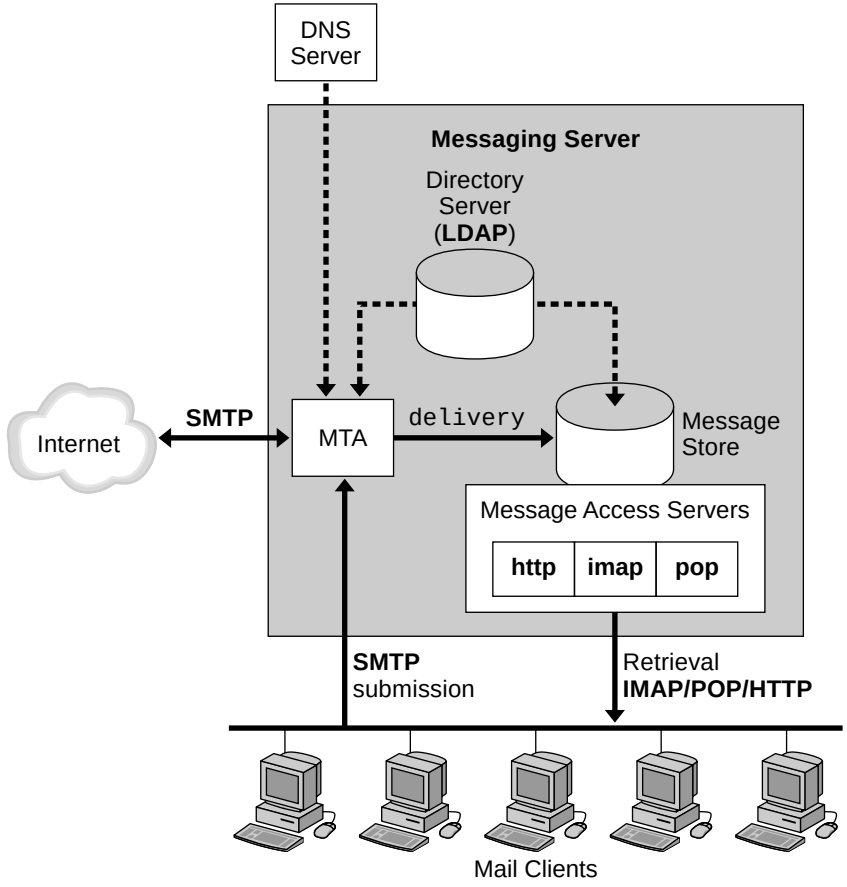
- Load balancing
- Firewalls
- High availability

These are also described in later sections.

Messaging Server Software Architecture

Figure 3-1 on page 35 shows a simplified standalone view of Messaging Server. While this particular deployment is not recommended because it does not scale well, it does illustrate the individual components of the server.

Figure 3-1 Standalone Messaging Server, Simplified Components View



The preceding figure shows the following Messaging Server software components:

- **Message Transfer Agent or MTA.** Receives, routes, transports, and delivers mail messages using the SMTP protocol. An MTA is like an electronic mail deliverer dispersing messages to an electronic mailbox or to another MTA.

- **Message Store.** Consists of a set of components that store, retrieve, and manipulate messages for mail clients. Mail can be retrieved by POP, IMAP, or HTTP clients. POP clients download messages to the client machine for reading and storage. IMAP and HTTP clients read and store messages on the server. The Message Store is like an electronic mailbox that stores and retrieves mail for users.
- **LDAP directory.** Stores, retrieves, and distributes mail directory information for Messaging Server. This includes user routing information, distribution lists, configuration data, and other information necessary to support delivery and access of email. The LDAP directory is a directory of mail addresses, aliases, routing information, passwords, and any other information needed by the MTA or Message Store to deliver and retrieve messages.
- **DNS Server.** Translates domain names into IP addresses. This component needs to be present before Messaging Server is installed.

Message Path Through the Simplified Messaging Server System

Incoming messages from the Internet or local clients are received by the MTA through the Simple Mail Transport Protocol (SMTP). If the address is internal, that is, within the Messaging Server domain, the MTA delivers the message to the Message Store. If the message is external, that is, addressed to a domain outside of Messaging Server control, the MTA relays the message to another MTA on the Internet.

Although it is possible to deliver mail to the `/var/mail` file system (UNIX systems only), which was the case in previous versions of the Messaging Server, local messages are usually delivered to the more optimized Messaging Server Message Store. Messages are then retrieved by IMAP4, POP3, or HTTP mail client programs.

The directory server stores and retrieves local user and group delivery information such as addresses, alternate mail addresses, and mailhost. When the MTA receives a message, it uses this address information to determine where and how to deliver the message.

In addition to storing messages, the Message Store uses the directory server to verify user login name and passwords for mail clients accessing their mail. The directory also stores information about quota limits, default message store type, and so on.

Outgoing messages from mail clients go directly to the MTA, which sends the message to the appropriate server on the Internet. If the address is local, the MTA sends the message to the Message Store.

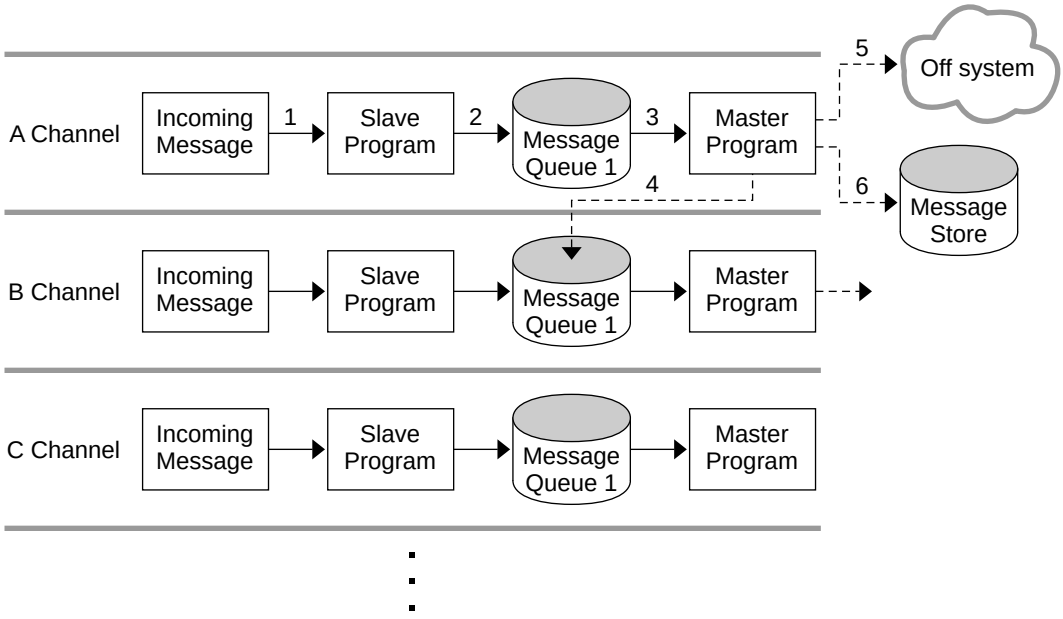
New users and groups are created by adding user and group entries to the directory. Entries can be created or modified by using the User Management Utility, or by modifying the directory using LDAP.

Messaging Server components are administered by the Administration Server console. In addition, Messaging Server provides a set of command-line interfaces and configuration files. Any machine connected to a Messaging Server host can perform administrative tasks (assuming, of course, the administrator has proper access). Some of the more common administrative tasks are adding, modifying, and removing users and groups to the mail system, and configuring the operation of the MTA, directory server, and Message Store.

The Message Transfer Agent (MTA)

The MTA routes, transfers, and delivers Internet mail messages for Messaging Server. Mail flows through interfaces known as *channels*, which consist of a pair of channel programs and a set of configuration information. Figure 3-2 on page 38 illustrates the process. You can configure channels individually and direct mail to specific channels based on the address.

Figure 3-2 Channel Architecture



Each channel consists of up to two channel programs called a *slave program*, which handles mail coming into the channel, and a *master program*, which handles mail as it leaves the channel. There is also an outgoing message queue for storing messages that are destined to be sent to one or more of the interfaces associated with the channel. Channel programs perform one of two functions:

- Slave programs (1) accept messages from other interfaces, (2) enqueue them into message queues for further processing by the MTA, or reject the message so it is not accepted onto the system.
- Master programs (3) process the message from the queue area, and (4) enqueue it on the same system for further processing by another channel, or (5) transmit messages off system to other interfaces, deleting them from their queue after they are sent, or (6) deliver the message to the final destination on the system, such as the Message Store.

Channels are configurable by using the `imta.cnf` configuration text file. Through channel configuration, you can set a variety of *channel keywords* to control how messages are handled. Channel keywords affect performance tuning as well as reporting aspects of the system. For example, you can define multiple channels to segment traffic by groups or departments, define message size limits to limit traffic, and define delivery status notification rules according to the needs of your

business. Diagnostic attributes are also configurable on a per-channel basis. The number of configuration parameters that can be set on a channel basis is large. See the *Sun Java System Messaging Server Administration Guide* for detailed information.

Messaging Server provides a number of default channels, including:

- **SMTP Channel.** Enables TCP/IP-based message delivery and receipt. Both master and slave channels are provided.
- **LMTP Channel.** Enables routing of messages directly from MTAs to the Message Store. These channels communicate with the Message Store over LMTP instead of SMTP.
- **Pipe Channel.** Used for alternative message delivery programs. Enables delivery of messages to programs such as a mail sorter rather than directly to a user's inbox.
- **Local Channel.** Delivers mail to `/var/mail`. Provides for compatibility with older UNIX mail clients.
- **Reprocessing Channel.** Useful for messages that are resubmitted.
- **Defragmentation Channel.** Reassembles partial messages into the original complete message to support the MIME message/partial content type.
- **Conversion Channel.** Performs body part by body part conversion on messages. Useful for rewriting addresses or re-formatting messages.

See the *Sun Java System Messaging Server Administration Guide* for more information on MTA concepts.

Direct LDAP Lookup

Prior to version 5.2, Messaging Server ran in `dirsync` mode. In `dirsync` mode, directory information about users and groups used by the MTA was accessed through a number of files and databases collectively called the directory cache. The data itself was stored in the LDAP directory, but actual information was accessed from the cache. Data in the cache was updated by the `dirsync` program, which monitored changes to the LDAP directory and updated the files and databases accordingly.

Starting with Messaging Server 5.2, you can configure the MTA to look up the information directly from the LDAP server. This direct lookup makes better use of LDAP by using the kind of normal query expected by an LDAP server. The direct lookup provides a more scalable, slightly faster, and more configurable

relationship between the MTA and the LDAP server. The results of the LDAP queries are cached in the process, with configurable size and time, so performance is tunable. See the *Sun Java System Messaging Server Administration Guide* for more information.

With the introduction of Messaging Server 6.0, `dirsync` mode is no longer supported nor included.

Rewrite Rules

Mail is routed to a channel based on the result of running the destination addresses through *domain rewriting rules*, or *rewrite rules* for short. Rewrite rules are used to convert addresses into true domain addresses and to determine their corresponding channels. These rules are used to rewrite addresses appearing in both the *transport layer* and the *message header*. The transport layer is the message's envelope. It contains routing information and is invisible to the user, but is the actual information used to deliver the message to the appropriate recipient.

The rewrite rules and the table of channels cooperate to determine the disposition of each address. The result of the rewrite process is a rewritten address and a routing system, that is, the system (channel) to which the message is to be sent. Depending upon the topology of the network, the routing system might only be the first step along the path the message takes to reach its destination, or it might be the final destination system itself.

After the rewrite process has finished, a search is made for the routing system among the channel portion of the `imta.cnf` file. Each channel has one or more host names associated with it. The routing system name is compared against each of these names to determine to which channel to enqueue the message. A simple rewrite rule is shown here:

```
example.com     $U%example.com@tcp_siroe-daemon
```

This rule matches addresses for the domain `example.com` only. Such matching addresses would be rewritten using the template `$U%$D`, where:

$U	Indicates the user portion or left-hand side of the address (before the @)
%	Indicates the @ sign
$D	Indicates the domain portion or right-hand side of the address (after the @)

Thus, a message of the form `wallaby@thor.example.com` would be rewritten to `wallaby@example.com`, and would be sent to the channel called `tcp_siroe-daemon`.

Rewrite rules can perform sophisticated substitutions based on mapping tables, LDAP directory lookups, and database references. While occasionally cryptic, they are useful in the fact that they operate at a low level and impose little direct overhead on the message processing cycle. For full information on these and other features available in the rewrite process, see the *Sun Java System Messaging Server Administration Guide*.

The Job Controller

The job controller controls master, job controller, sender, and dequeue channel programs. The job controller is a program that controls the message queues and executes the programs to do the actual message delivery. The job controller runs as a multithreaded process and is one of the few processes that is always present in the Messaging Server system. The channel processing jobs themselves are created by the job controller but are transient and might not be present when there is no work for them to do.

There are configurables for the job controller that determine if there is always at least one instance of a channel processing program. In many cases, these are set so that there is always at least one instance of the service program even when there is no immediate work to do. In other cases, there will be an instance for a set period of time after it last did some work but there is nothing left to do.

Slave channels, which respond to external stimuli, notify the job controller of a newly created message file. The job controller enters this information into its internal data structure and if necessary creates a master channel job to process the message. This job creation might not be necessary if the job controller determines that an extant channel job can process the newly created message file. When the master channel job starts, it gets its message assignment from the job controller. When it is finished with the message, the master channel updates the job controller as to the status of its processing. The status is either that the message is successfully dequeued or the message should be rescheduled for retrying. The job controller maintains information about message priority and previous delivery attempts that failed, allowing for advantageous scheduling of channel jobs. The job controller also keeps track of the state of each job. The state can be idle, how long the job has been idle, or whether the job is busy. Tracking state enables the job controller to keep an optimal pool of channel jobs.

Local Mail Transfer Protocol (LMTP)

As of the Sun ONE Messaging Server 6.0 release, you can configure LMTP for delivery to the Message Store in a multi-tier deployment. In these scenarios, where you are using inbound relays and back-end Message Stores, the relays become responsible for address expansion and delivery methods such as autoreply and forwarding and also for mailing list expansion. Delivery to the back-end stores historically has been over SMTP, which requires the back-end system to look up the recipient addresses in the LDAP directory again, thereby engaging the full machinery of the MTA. For speed and efficiency, the MTA can use LMTP rather than SMTP to deliver messages to the back-end store. See the *Sun Java System Messaging Server Administration Guide* for more information.

NOTE Messaging Server's implementation of LMTP is a general purpose implementation. You can only use Messaging Server's LMTP between the Messaging Server MTA and Message Store components in a two-tier architecture. In a single-tier architecture, consisting of one machine, you cannot use LMTP.

The Message Store

The Message Store is a dedicated data store for the delivery, retrieval, and manipulation of Internet mail messages. The Message Store works with the IMAP4 and POP3 client access servers to provide flexible and easy access to messaging. The Message Store also works through the Webmail server to provide messaging capabilities to web browsers. In addition to this section, see the *Sun Java System Messaging Server Administration Guide* for more information.

The Message Store is organized as a set of folders or user mailboxes. The folder or mailbox is a container for messages. Each user has an INBOX where new mail arrives. Each user can also have one or more folders where mail can be stored. Folders can contain other folders arranged in a hierarchical tree. Mailboxes owned by an individual user are private folders. Private folders can be shared at the owner's discretion with other users on the same Message Store. As of the 6.0 release, Messaging Server supports sharing folders across multiple stores.

There are two general areas in the Message Store, one for user files and another for system files. In the user area, the location of each user's INBOX is determined by using a two-level hashing algorithm. Each user mailbox or folder is represented by another directory in its parent folder. Each message is stored as a plain text file using the MIME formatting standard. When there are many messages in a folder, the system creates hash directories for that folder. Using hash directories eases the

burden on the underlying file system when there are many messages in a folder. In addition to the messages themselves, the Message Store maintains an index and cache of message header information and other frequently used data to enable clients to rapidly retrieve mailbox information and do common searches without the need to access the individual message files.

A Message Store can contain many message store partitions. A Message Store partition is contained by a file system volume. As the file system becomes full, you can create additional file system volumes and Message Store partitions on those file system volumes.

Message Store maintains only one copy of each message per partition. This is sometimes referred to as a single-copy message store. When the Message Store receives a message addressed to multiple users or a group or distribution list, it adds a reference to the message in each user's INBOX. Rather than saving a copy of the message in each user's INBOX, Message Store avoids the storage of duplicate data. The individual message status flag (seen, read, answered, deleted, and so on) is maintained per folder for each user.

The system area contains information on the entire Message Store in Berkeley database format for faster access. The information in the system area can be reconstructed from the user area. Starting with Sun ONE Messaging Server 5.2, the product contains a database snapshot function. When needed, you can quickly recover the database to a known state. Messaging Server now also adds fast recovery, so that in case of database corruption, you can shut down the Message Store and bring it back immediately without having to wait for a lengthy database reconstruction.

The Message Store supports the IMAP quota extension (RFC2087). Enforcement of quota can be turned on or off. You can configure a user quota by using number of bytes or number of messages. You can also set a threshold so that if the quota reaches the threshold, a warning message can be sent to the user. When the user is over quota, new messages can be held up for retry during a grace period. After the grace period, messages sent to the over-quota user are returned to the sender with a non-delivery notification.

For special applications where quota is used, but messages must be delivered regardless of the quota status of the users, there is a guaranteed message delivery channel. This channel can be used to deliver all messages regardless of quota status. Utilities are available for reporting quota usage and for sending over quota warnings.

Directory Services

Messaging Server is bundled with Sun Java System Directory Server. Directory Server is a Lightweight Directory Access Protocol (LDAP) directory service. Directory Server provides the central repository for information critical to the operation of Messaging Server. This information includes user profiles, distribution lists, and other system resources.

Directory Information Tree

The directory stores data in the form of a tree, known as the Directory Information Tree (DIT). The DIT is a hierarchical structure, with one major branch at the top of the tree and branches and subbranches below. The DIT is flexible enough to enable you to design a deployment that fits your organization's needs. For example, you might choose to arrange the DIT according to your actual business organizational structure, or by the geographical layout of your business. You also might want to design a DIT that follows a one-to-one mapping to your DNS layers. Use care when designing your DIT, as changing it after the fact is not an easy task.

The DIT is also flexible enough to accommodate a wide range of administration scenarios. You can administer the DIT in either a centralized or distributed manner. In centralized administration, one authority manages the entire DIT. You would use centralized administration where the entire DIT resides on one mail server. In distributed administration, multiple authorities manage the DIT. Usually you implement distributed administration when the DIT is divided into portions, or subtrees, residing on different mail servers.

When the DIT is large, or when mail servers are geographically disbursed, consider delegating management of portions of the DIT. Typically, you assign an authority to manage each subtree of the DIT. Messaging Server enables you to manage multiple subtrees from one authority. However, for security reasons, an authority can only make changes to the subtree of the DIT that the authority owns.

The default schema used by Messaging Server when Identity Server is not used is different from the one used by Identity Server. Messaging Server supports both Sun Java System LDAP Schema 1 and 2, and allows for transition and migration of the schemas. See Chapter 7, "Understanding Messaging Server Schema and Provisioning Options" for more information.

Directory Replication

Directory Server supports replication, enabling a variety of configurations that provide redundancy and efficiency. Enabling replication of all or part of the DIT from one host to other hosts provides the following configuration capabilities:

- The directory information is more accessible, because it is replicated on multiple servers rather than on a single server.
- The directory information can be cached on a local directory server, reducing effort of accessing the information from a remote directory server. Caching the directory information enhances performance, especially in deployments with limited network bandwidth to the central directory.
- Depending on the actual configuration, multiple directory servers can process mail client requests faster than a single centralized server.

For more information on directory replication, directory performance tuning, and DIT structure and design, see the Sun Java System Directory Server documentation at the following location:

`http://docs.sun.com/db/coll/DirectoryServer_04q2`

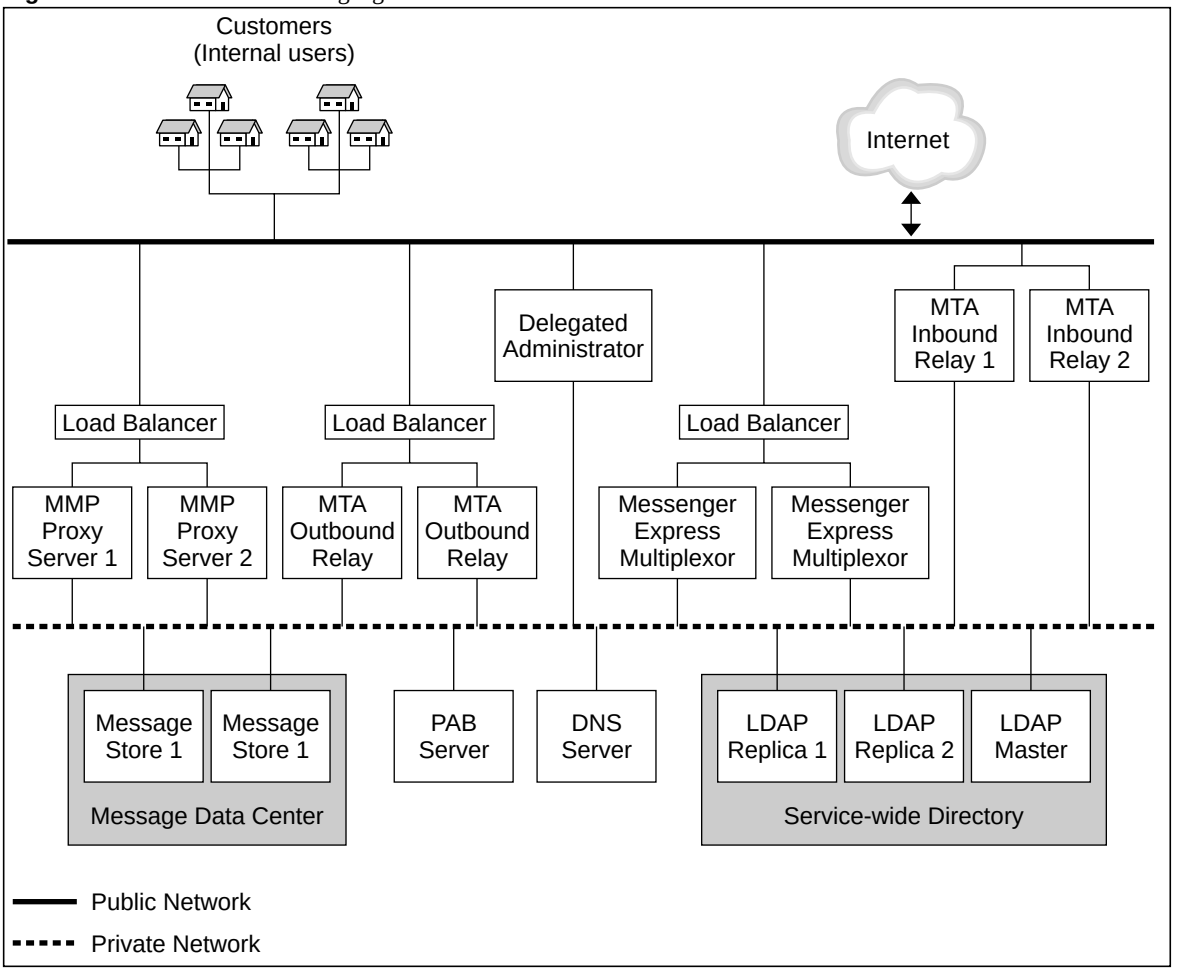
Understanding the Two-tier Architecture

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

Figure 3-3 on page 46 shows a simplified two-tier architecture.

Figure 3-3 Two-Tier 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 personal address book (PAB) 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 3-3 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. In this instance, you retrieve user and group data from the existing directory and place it in a Sun Java System Directory Server directory. The data format of the existing directory must also be compliant with the Messaging Server schema.

Message Store. Holds and stores user mail. Also referred to as a "back end." The Message Store also refers to the Message Access Components such as the IMAP server, the POP server, and the Messenger Express (Webmail) servers. Figure 3-3 on page 46 shows a deployment that has two message stores. You can add more stores as needed.

Personal Address Book (PAB) Server. Stores and retrieves Messenger Express user addresses.

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 line. You must install an operating DNS server prior to installing Messaging Server.

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 3-3 on page 46 has two load balancers. One balances connections to the MMP, and one balances the MTA Outbound Relays. Load balancers are not part of the Java Enterprise System product line. You cannot use load balancers on the Message Store or directory masters. You use them for connections to MMPs, MEMs, Communications Express, inbound and outbound MTAs, directory consumers, and without Messaging Server's MTA's use of the Brightmail product, Brightmail servers.

MTA Inbound Relay. MTA dedicated to accepting messages from external (Internet) sites and routing those messages to 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.

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.

There is another option for routing that outbound relays never deliver internally. They view internally bound mail from their user base as simply an instance of routing and forward all such messages to an inbound MTA.

Delegated Administrator Server. Provides a GUI management console for users and administrators. Delegated Administrator enables users to change passwords, set vacation mail, and so forth. Administrators are able to do more advanced administrative tasks, such as adding and deleting users. Delegated Administrator currently works only with Schema 1 implementations.

Messaging Multiplexor or *Mail Message Proxy* 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 DNS 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 proxy 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.

Messenger Express Multiplexor. A specialized server that acts as a single point of connection to the HTTP access service for Webmail. All users connect to the single messaging proxy server, which directs them to their appropriate mailbox. As a result, an entire array of messaging servers will appear to mail users to be a single host name. While the Messaging Multiplexor (MMP) connects to POP and IMAP servers, the Messenger Express Multiplexor connects to an HTTP server. In other words, the Messenger Express Multiplexor is to Messenger Express as MMP is to POP and IMAP.

Two-tier 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-tier 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 if the message is addressed to an external user or an internal user. If it is an external user, it sends the message to the Internet. If it is an internal user, it sends it 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 / MEM / Communications Express Proxy Server 1 or 2 -> Message Store 1 or 2

Mail is retrieved by using either POP, HTTP, or IMAP. The user connection is received by the load balancer and forwarded to one of the MMP, MEM, or Communications Express 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 mediates for the rest of the connection between the client and servers. The exception is for Communications Express, which does a level of processing of ongoing user requests to handle some of the browser rendering.

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, (or directly to the appropriate message store if so configured). The outbound relay reads the address and determines if the message is addressed to an external user or an internal user. If it is an external user, it sends the message to an MTA on the Internet. If it is an internal user, it sends it to MTA Inbound Relay 1 or 2. The MTA Inbound Relay delivers the message to the appropriate Message Store. The Message Store receives the message and delivers it to the appropriate mailbox.

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 if the message is addressed to an external user (if Internet relaying is enabled) or an internal user. If it is an external user, the inbound relay sends the message to another MTA on the Internet. If it is 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

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 growths 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 begin to maximize resources of your existing architecture. Careful planning helps you to determine how to appropriately scale your architecture.

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

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

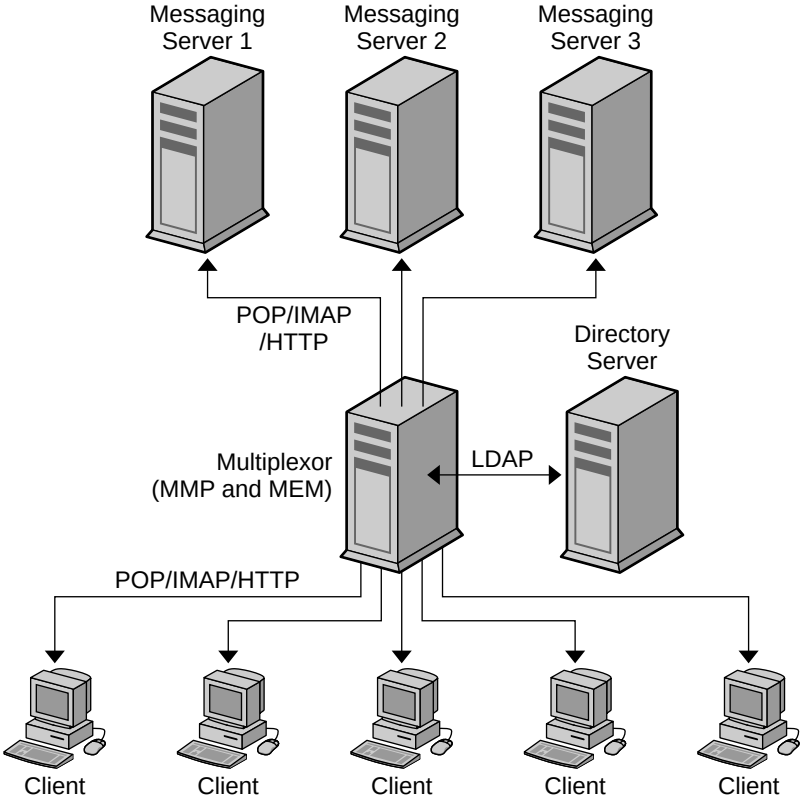
Spreading Your User Base Across Several Servers

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.

The Messaging Multiplexor (MMP) is a multi-threaded server that handles incoming client connections for multiple servers. The MMP accepts POP or IMAP connections, performs LDAP lookups for authentication, and then routes connections to the appropriate messaging server. For HTTP connections, you might enable the Messenger Express Multiplexor (MEM) to handle incoming client connections for multiple servers. Communications Express also acts in a similar fashion.

Often, both the MMP and the Messenger Express Multiplexor are placed on the same machine for ease of manageability. Figure 3-4 on page 52 shows a sample architecture where users are spread across multiple back-end servers and a multiplexor is enabled to handle incoming client connections.

Figure 3-4 Spreading Your User Base Across Multiple Servers



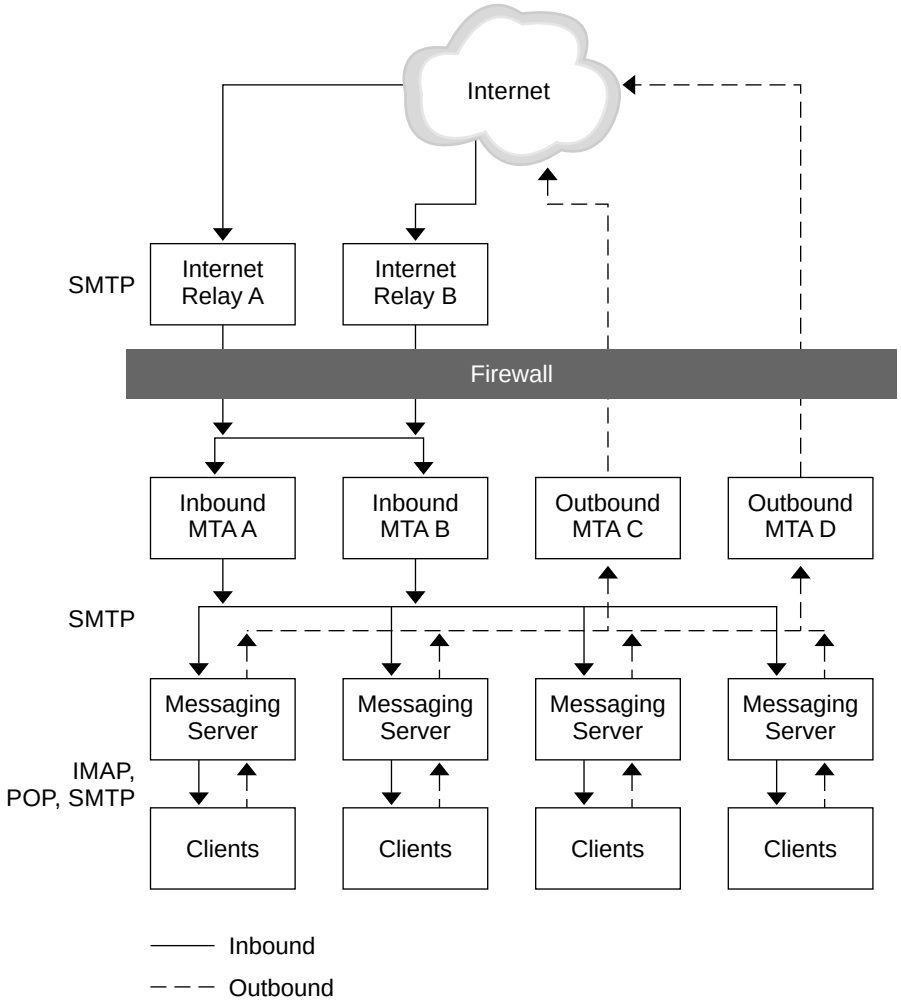
Spreading users across back-end servers provides simplified user management, as long as you use the MMP or the MEM. Because users connect to one back-end server, where their mail resides, you can standardize setup across all users. This configuration also makes administration of multiple servers easier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

Spreading Your Resources Across Redundant Components

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

The following figure is an example of spreading resources across redundant MTA relays. The same set of components, such as the Internet relay, inbound MTA, and outbound MTA, are used in Figure 3-4 on page 52, except that in this case, there are two of each deployed.

Figure 3-5 Spreading Your Resources Across Redundant Components



By using redundant MTA relays, you can ensure that if one component is disabled, the other is still available. Also, spreading resources across redundant MTA relays enables load sharing. For example, two Internet relays share the load that a single relay previously managed. 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 mail relays 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 MX Records, Relays, and Load Balancers is described in the following sections.

MX Records

Equal priority MX records route messages to redundant Internet relays and inbound and outbound MTAs. For example, the sending MTA will find that the MX record for `siroe.com` corresponds to `relayA.siroe.com` and `relayB.siroe.com`. One of these relays is chosen at random, as they have equal priority, and an SMTP connection is opened. If the first relay chosen does not respond, the mail goes to the other relay. See the following MX record example:

```
siroe.com in MX 10 relayA.siroe.com
siroe.com in MX 10 relayB.siroe.com
```

Relays

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 mail relays. You can further share the load by designating different relays to handle outgoing and incoming messages.

Often, both the inbound and outbound relays are combined as a single In / Out SMTP relay host. To determine if you need one or more relay 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 Architectural Strategies" in Chapter 6, "Planning Your Sizing Strategy."

- Test the load of a prototype of your system.

  See "Using a Load Simulator"in Chapter 6, "Planning Your Sizing Strategy."

- Monitor system performance and adjust the deployment accordingly.

Planning for High Availability

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 both the Sun™ Cluster services 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.

See Chapter 10, “Planning for Service Availability” for more information.

Performance Considerations for a Messaging Server Architecture

This section describes how to evaluate the performance characteristics of Messaging Server components to accurately develop your architecture.

This section contains the following sections:

Message Store Performance Considerations

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

1. Disk I/O
2. Inbound message rate (also known as message insertion rate)
3. Message sizes
4. Login rate (POP/IMAP/HTTP)
5. Transaction rate for IMAP and HTTP
6. Concurrent number of connections for the various protocols
7. Network I/O

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.

Messaging Server Directories

Messaging Server uses five directories that receive a significant amount of input and output activity. Because these directories are accessed very frequently, you can increase performance by providing each of those directories with its own disk, or even better, providing each directory with a Redundant Array of Independent Disks (RAID). The following table describes these directories.

Table 3-1 High Access Messaging Server Directories

High I/O Directory	Description and Defining Parameter
MTA queue directory	In this directory, many files are created, one for each message that passes through the MTA channels. After the file is sent to the next destination, the file is then deleted. The directory location is controlled by the `IMTA_QUEUE` option in the `imta_tailor` file. Before modifying the MTA queue directory, read about this option in the *Sun Java System Messaging Server Administration Reference*. Default location: *msg_svr_base*`/data/imta/queue`
Messaging Server log directory	This directory contains log files which are constantly being appended with new logging messages. The number of changes will depend on the logging level set. The directory location is controlled by the `configutil` parameter `logfile.*.logdir`, where * can be a log-generating component such as `admin`, `default`, `http`, `imap`, or `pop`. The MTA log files can be changed with `IMTA_LOG` option in the `imta_tailor` file. Default location: *msg_svr_base*`/data/log`
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 *msg_svr_base*`/data/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 `configutil` parameter `store.partition.*.path`, 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: *msg_svr_base*`/data/store/partition/primary`

Table 3-1 High Access Messaging Server Directories *(Continued)*

High I/O Directory	Description and Defining Parameter
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 `configutil` parameter `store.partition.*.messagepath`, where * is the name of the partition. Some sites might have a single message store partition called `primary` specified by `store.partition.primary.path`. Large sites might have additional partitions that can be specified with `store.partition.*.path`, where * is the name of the partition. Default location: *msg_svr_base*`/data/store/partition/primary`

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

MTA Queue Directories

In non-LMTP environments, the MTA queue directories are also heavily used. LMTP 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. If the system is standalone or uses the local MTA for Webmail sends, significant I/O can still result on this directory for outbound mail traffic. In a proper two-tier 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.

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 LUN for logging. For very large store deployments, or environments where significant logging is required, a dedicated LUN is in order.

In almost all environments, you need to protect the Message Store from loss of data. The level of loss and continuous availability that is necessary varies from simply disk protection such as RAID5, 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.

mboxlist Directory

The `mboxlist` directory is highly I/O intensive but not very large. The `mboxlist` directory contains the Sleepycat (Berkeley) databases that are used by the stores and their transaction logs. Because of its high I/O activity, and due to the fact that it cannot be split, 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 Sleepycat 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

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, `fsck` 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, re-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. (By way of comparison, a small drive in a Sun disk array would be 36 GB, a large drive would be 73 GB or greater.) Again, every deployment is different and needs to examine its own set of requirements.

Message Store Scalability

The Message Store scales well, due to its multiprocess, multithreaded nature. The Message Store actually scales more than linearly from one to four processors, meaning 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.

MTA Performance Considerations

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

- 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 need to do spam or virus filtering

- The use of SIEVE rules and the need to do other message parsing (like use of the conversion channel)

The MTA router is both CPU and I/O intensive. The MTA uses two different file systems for the queue directory and the logging directory. For a small host (four processors or less) functioning as an MTA router, 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.

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 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 Scalability

The MTA router 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 router 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 MTA routers are offline for some reason.

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

Mail Message Proxy (MMP) Performance Considerations

The MMP uses no disk I/O other than for logging. The MMP is completely CPU and network bound. Unlike all the other Messaging Server components, the MMP is not multiprocess and multithreaded. The primary execution code is single process and multithreaded. Thus, because the MMP is not sufficiently a multiprocess, it does not scale as well as the other components.

The MMP does not scale beyond four processors, and scales less than linearly from two to four processors. Two processor, rack mounted machines are good candidates for MMPs.

In deployments where you choose to put other component software on the same machine as the MMP (MEM, Calendar Server front end, Communications Express Web Client, LDAP proxy, and so on), look at deploying a larger, four processor SPARC machine. Such a configuration reduces the total number of machines that need to be managed, patched, monitored, and so forth.

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.

If you use SSL in a significant percentage of connections, install a hardware accelerator.

MMP and High Availability

Never deploy the MMP under HA control. An individual MMP has no static data. In a highly available environment, add one or more additional MMP machines so that if one or more are down there is still sufficient capacity for the peak load. If you are using Sun Fire Blade™ Server hardware, take into account the possibility that an entire Blade rack unit can go down and plan for the appropriate redundancy.

Messenger Express Multiplexor (MEM) Performance Considerations

The MEM provides a middle-tier proxy for the Webmail client. This client enables users to access mail and to compose messages through a browser. The benefit of the MEM is that end users only connect to the MEM to access email, regardless of which back-end server is storing their mail. MEM accomplishes this by managing the HTTP session information and user profiles via the user's LDAP information. The second benefit is that all static files and LDAP authentication states are located on the Messaging Server front end. This benefit offsets some of the additional CPU requirements associated with web page rendering from the Message Store back end.

The MEM has many of the same characteristics as the MMP. The MEM will scale beyond four processors, but in most environments, there is no particular value in doing so. Also, in the future, the Webmail component will be offloaded from the Message Store and onto access layer machines that are running the XML rendering as Java servlets under the web server. Java servlets do not presently scale well beyond two processors. Thus, plan your hardware choice around either SPARC or Intel two-processor machines for the MEM, or assume that you will repurpose your current two-processor MEM hardware to be replaced by smaller machines when the next generation solution becomes available.

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

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).

Setting the Mailbox Database Cache Size

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 `configutil` parameter `store.dbcachesize`.

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 handled by cache. An optimally sized cache will have a 99 percent hit rate (that is, 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 cache such that it holds a number of pages such that the cache will be able to return at least 95 percent of the requested data. If the direct cache return is less than 95 percent, then you need to increase the cache size.

The Sleepycat database command `db_stat` can be used to measure the cache hit rate. For example:

```
# /opt/SUNWmsgsr/lib/db_stat -m -h /var/opt/SUNWmsgsr/store/mboxlist

2MB 513KB 604B  Total cache size.
1               Number of caches.
2MB 520KB       Pool individual cache size.
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. (A cache that is too large will always show 99 percent.) The way to test this is to 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 `db_stat` shows a hit rate of less than 95 percent, then you should increase the cache size with `store.dbcachesize`.

NOTE As your user base changes, the hit rate can also change. Periodically check and adjust this parameter as necessary. This parameter has an upper limit of 2 GB imposed by the Sleepycat database.

Chapter 4

Determining Your 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 goals ensures that you will have an architecture that can scale and grow.

This chapter contains the following sections:

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:

1. Physical communication links, such as cable length, grade, and so forth
2. Communication links, such as analog, ISDN, VPN, T3, and so forth, and available bandwidth and latency between sites
3. Server information, including:
    - Host names
    - IP addresses

- Domain Name System (DNS) server for domain membership

4. Locations of devices on your network, including:
   - Hubs
   - Switches
   - Modems
   - Routers and bridges
   - Proxy servers
5. 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
- Firewalls
- Load balancers
- Storage Area Network (SAN)
- 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.

Firewalls

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 Solaris 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 Solaris™ Security Toolkit provides a flexible and extensible mechanism to minimize, harden, and secure Solaris Operating Environment 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.

Storage Area Networks (SANs)

Understanding the data requirements of the storage system is necessary for a successful deployment. Increasingly, SANs are being deployed so that the storage is independent of the servers used in conjunction with it. Deploying SANs 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?

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 need to consider the following perspectives:

- DMZ
- Intranet
- Internal Network
- Proxies

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. To limit exposure in case of a security breach from such attacks, these servers typically contain no information about the internal network. For example, the nameserver facilities only include the server and the routers to the Internet.

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.

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.

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.

These machines 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.

Proxies

Only the machines directly communicating with machines on the Internet should reside in the DMZ. If users require Internet access, though, this creates a problem based on your previous topology decisions. In this situation, proxies become helpful. Place a proxy on an internal network segment, or, better yet, an intranet segment. A machine requiring access to the Internet can pass its request onto the proxy, which in turn makes the request on the machine's behalf. This relay out to the Internet helps shield the machine from any potential danger it might encounter.

Because the proxy communicates directly with machines on the Internet, it should reside in the DMZ. However, this conflicts with the desire to prevent internal machines from directly communicating with DMZ machines. To keep this communication indirect, use a double proxy system. A second proxy residing in the intranet passes connection requests of the internal machines to the proxy in the DMZ, which in turn makes the actual connection out on the Internet.

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.

Chapter 5

Designing a Messaging Topology

The architecture design provides for how Messaging Server components are distributed across hardware and software resources, which provides the basis for determining requirements for the deployment environment design.

This chapter describes 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.

This chapter contains the following sections:

Identifying Your Geographic Needs

The first step in designing your messaging topology is to identify your geographic needs. In particular, you need to 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

Determining a Topology Design Strategy

Before you develop your topology, you need a strategy to determine where you are going to put your messaging servers in your enterprise. 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.

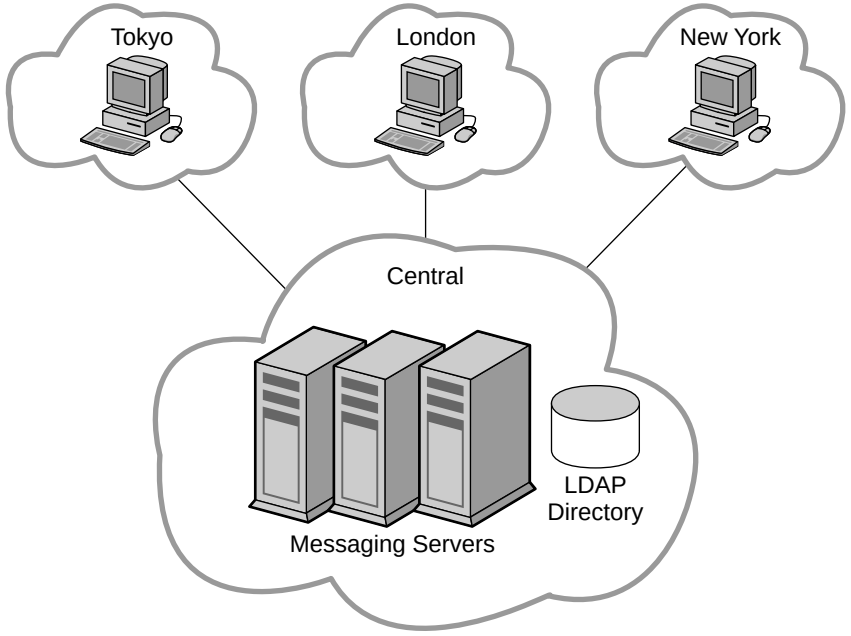
- Service Provider Topology

  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 on page 77 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).

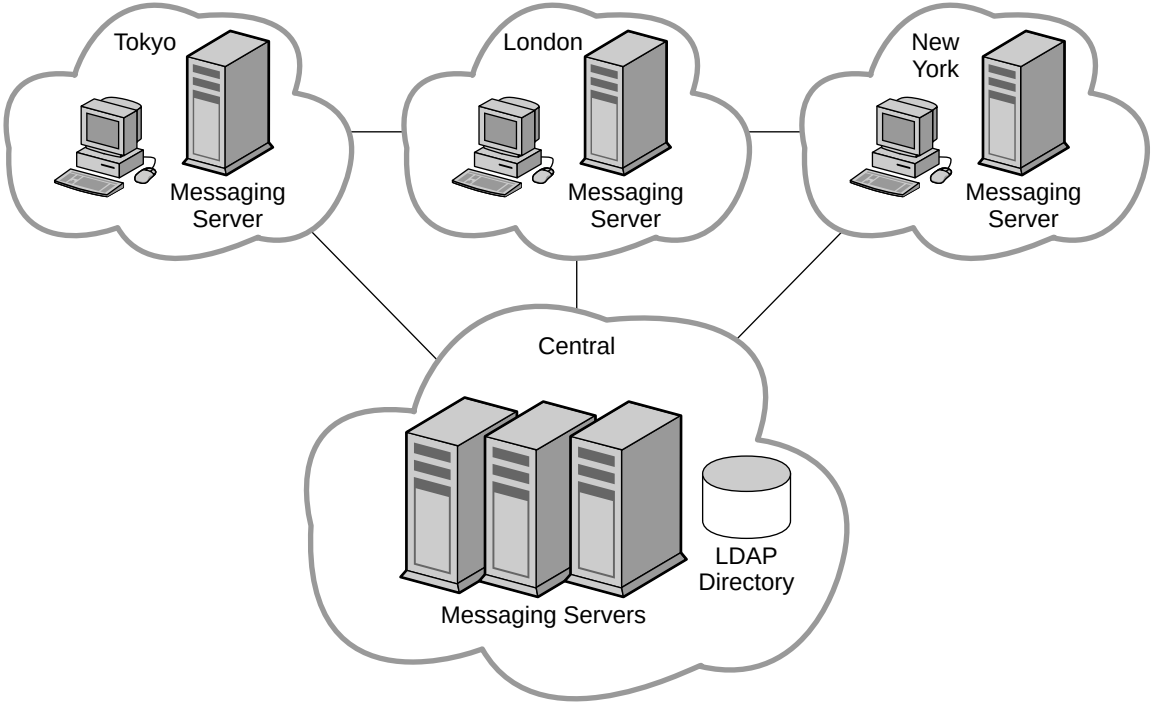
There are advantages to implementing a central topology. In general, a central topology has lower hardware and support costs. Central topologies tend to be easier to manage because you have a simplified messaging architecture and a directory replication structure with fewer replication agreements. With 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 on page 77, 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. The following figure 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.

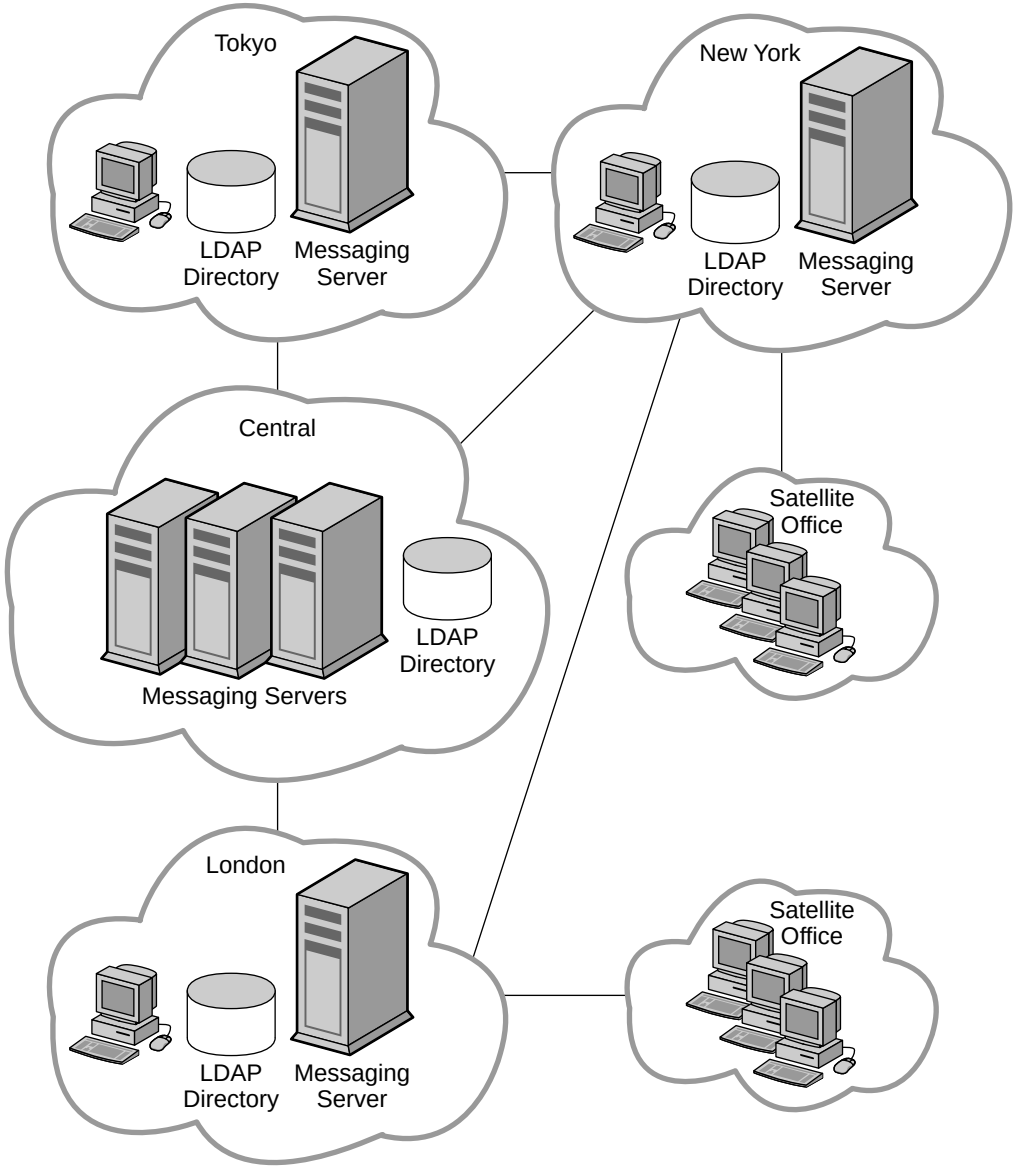
There are advantages to implementing a distributed topology. Users at regional sites have faster access to their messages because they do not have to retrieve messages over the WAN. Furthermore, messages sent within a regional location will have less messaging traffic than in a central topology. However, 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 Messaging Server because there are multiple servers spread across multiple sites.

Hybrid Topology

In a hybrid topology, central and distributed topologies are combined to meet the needs of an organization. Figure 5-3 on page 80 shows a hybrid topology.

Figure 5-3 Hybrid Topology

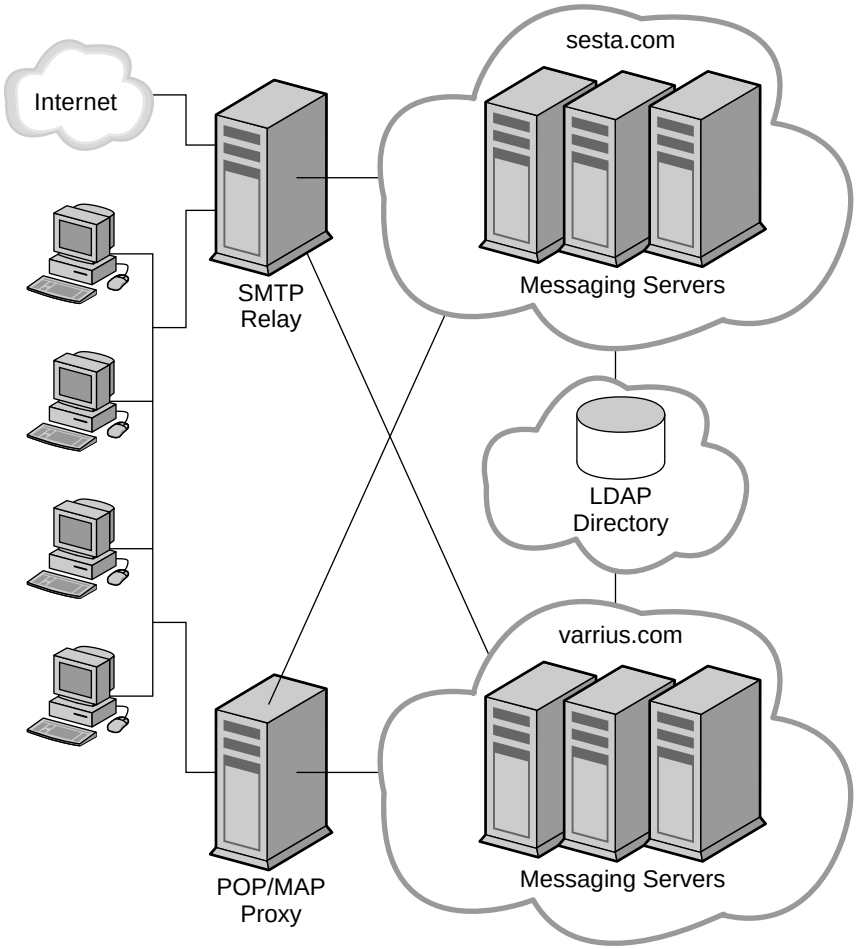


Organizations that benefit from a hybrid topology include those with many sites that have the ability to support a large user base. These sites that support them can house their own messaging servers. Some of these larger sites might have smaller satellite offices located in the general vicinity. But 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 on page 82 shows a service provider topology.

Figure 5-4 Service 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
- Mail Relays

Messaging Topology Components

In "Determining a Topology Design Strategy" on page 76, 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 Relay** 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 appropriate Messaging servers for retrieving messages.

Internet Relay. Routes messages from the Internet and relays them across the firewall. Typically, a Messaging server is set up to perform this function.

MTA Relay. The incoming 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 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.

Mail Relays

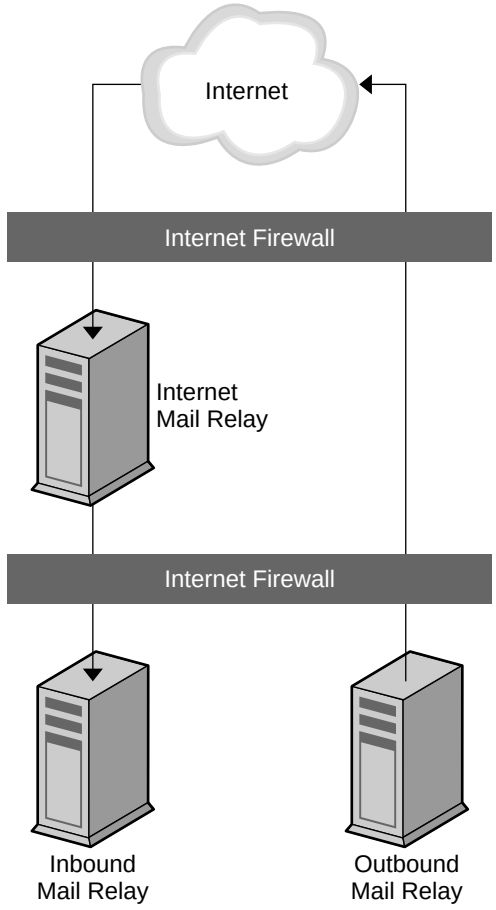
This section describes how you can use mail relays to protect your Messaging system as well as to control the flow of message traffic to and from your site.

An Internet relay is a single point of contact that receives messages from sites external to your organization. An Internet relay 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 relay 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 on page 85 illustrates the idea.

Figure 5-5 Mail Relays in Messaging Topology

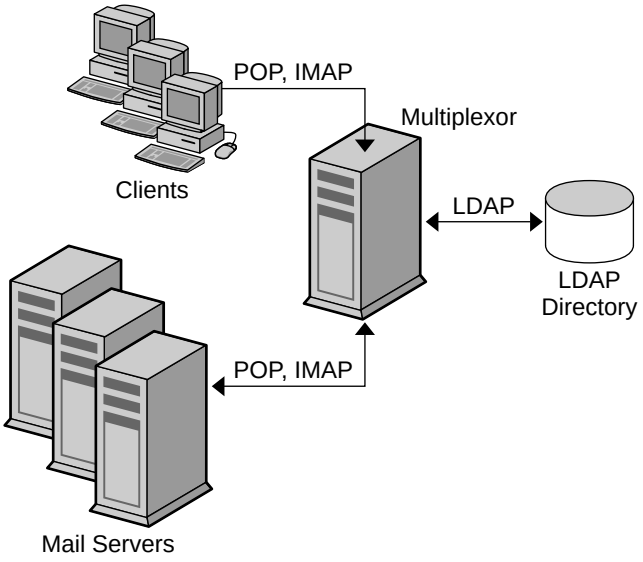


Messaging Multiplexor (MMP) and Messenger Express Multiplexor (MEM)

The MMP enables you to mask the layout of your Messaging servers from your end users. Consequently, you assign users to a generic MMP 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. The following figure shows how the MMP acts as a proxy for IMAP4 and POP3 connections to Messaging servers. You can multiplex HTTP services (like Messenger Express) by enabling the MEM feature. The following figure shows how multiplexors function in a Messaging Server environment.

Figure 5-6 Multiplexor Overview



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.

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 help you create a messaging topology, this section uses the example of the Siroe Corporation.

The Siroe Corporation is a multi-media 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. Similar to Chapter 2, "Analyzing Your Requirements," Siroe's messaging goals can be categorized into business objectives, technical, and financial constraints.

Siroe'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 Siroe 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 messaging servers 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.

Siroe's Financial and Technical Constraints

Because of budgetary restrictions, Siroe 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.

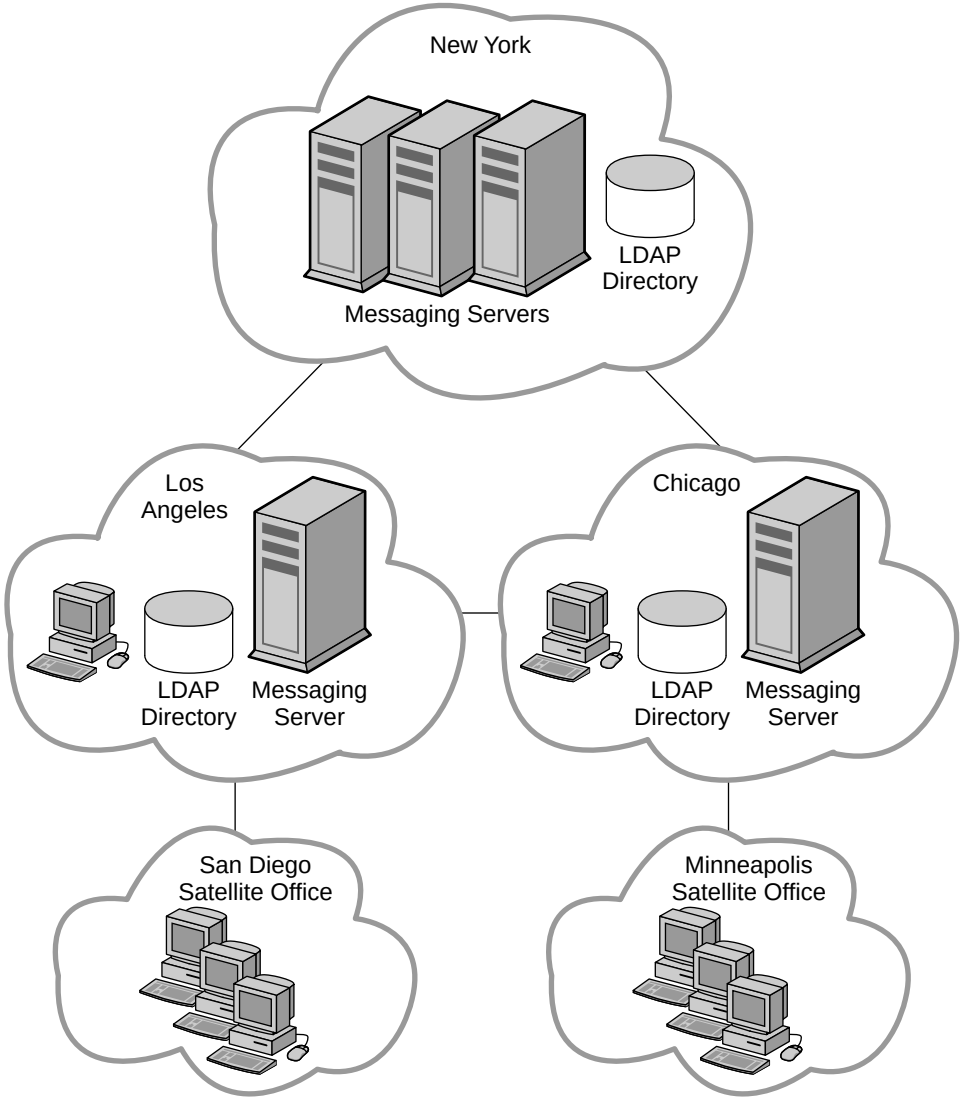
Step 2: Choosing Your Topology Strategy

The second step in creating your messaging topology is to choose your topology strategy, described in "Determining a Topology Design Strategy" on page 76. The Siroe Corporation evaluated their business objectives as well as their financial and technical constraints. They determined:

- Messaging servers 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 Siroe Corporation then mapped their objectives and constraints to a common design strategy. Figure 5-7 on page 89 shows that the Siroe Corporation has chosen a hybrid topology.

Figure 5-7 Hybrid Topology for the Siroe Corporation

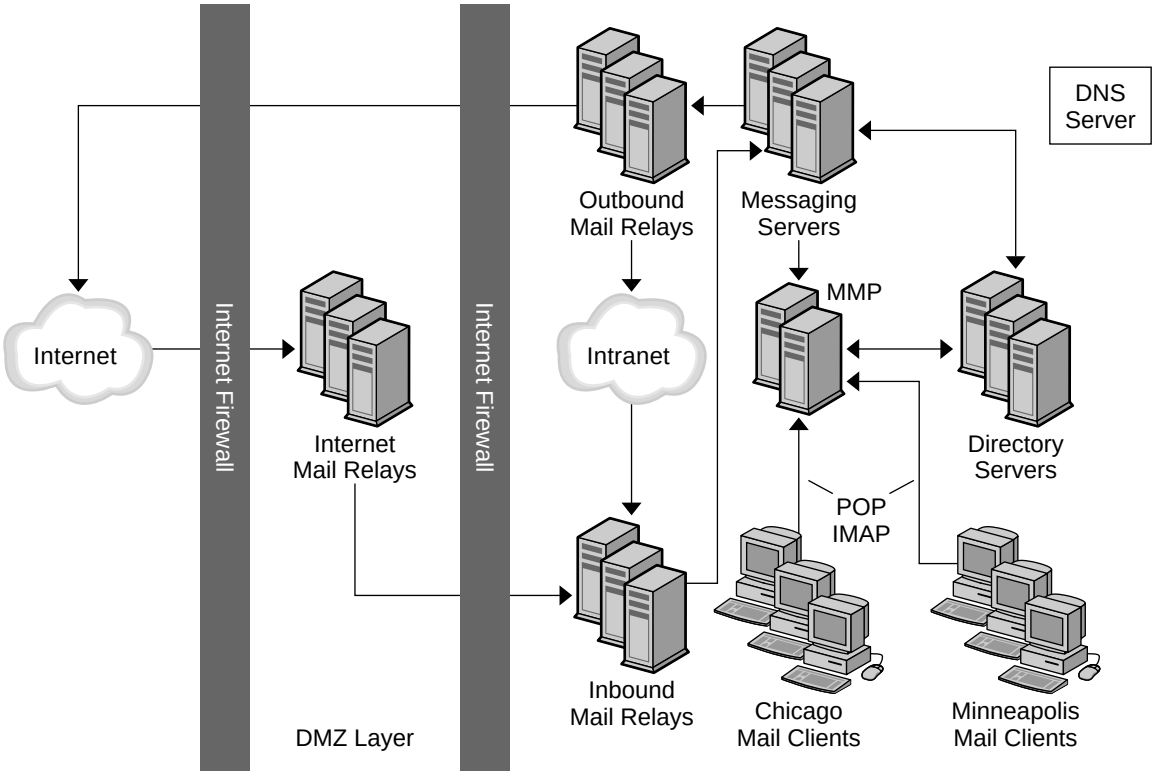


Because New York has the highest message transaction rate of messages entering and leaving the system, it has the most number of messaging servers. The smaller offices, Los Angeles and Chicago, also support San Diego and Minneapolis. But these satellite offices do not require their own messaging servers.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

Step 3: Planning Your 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" on page 82. The following figure illustrates the topology elements in the Chicago and Minneapolis offices.

Figure 5-8 Topological Elements in the Siroe 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 relays are placed in the topology to route messages from the Internet and relay them across the firewall. MTA relays 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 don't 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.

Chapter 6

Planning Your Sizing Strategy

When you design your deployment, you must decide how to configure your 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 introduces the basics of sizing your Messaging 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.

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 your Sun technical representative for your deployment hardware and software needs.

The chapter contains the following sections:

Collecting 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 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 to 10 in the morning, 12 to 1 in the afternoon, and 5 to 6 in the evening.

Consider the following points when analyzing peak volume:

1. Determine when and for how long the peaks occur.
2. Size 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. When you determine the peak volume on your system, be sure that your Messaging Server deployment can support it.

Creating Your 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 login 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 in the following table.

Table 6-1 Active Versus Inactive User

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 are not the same as mail attributes with `active` status, such as `mailuserstatus` or `inetuserstatus`. For more information on mail attributes, see the *Sun Java System Communications Services 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 your Professional Services representative to determine your individual needs.

2. How many connections are on your system during your peak volume for your POP, IMAP, and Messenger Express client access services?

   Specifically, note the number of concurrent, idle, and busy connections for each client access service that you support. Table 6-2 on page 96 defines these terms.

Table 6-2 Connections on Client Access Services

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 Messenger Express client can only have one connection per client. Furthermore, because POP and Messenger Express 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 Messenger Express 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:

a. Count the number of established TCP connections by using the `netstat` command on UNIX platforms.

b. Obtain the last login and logout times for users for Messenger Express or for IMAP client access services. See the *Sun Java System Messaging Server Administration Guide* for more information.

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.

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, and the rate of message retrieval.

8. Will you be using Secure Sockets Layer (SSL)? 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. 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.

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?

   For example, you might consider high availability.

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?

Defining Your 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 Messenger Express

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 Load Simulator" on page 100).

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 Kbyte or smaller in size, and about 20 percent of messages are 10 Kbyte 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 Kbyte or smaller in size. Thirty percent of messages in this user base are between 5-10 Kbyte. Five percent of messages are larger than 1 Mbyte. Of these users, 85 percent delete all of their messages after reading them. But, 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 send 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 Messenger Express

A mediumweight Messenger Express user base is similar to Mediumweight IMAP users. 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 Load Simulator."

Using a Load Simulator

To measure the performance of your Messaging Server, use your user bases (described in "Defining Your User Base" on page 98) and your usage profile (described in "Creating Your Usage Profile" on page 94) 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 Sun Professional Services for recommended load simulators and support.

Assessing Your 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:

- Memory Utilization
- Disk Throughput
- Disk Capacity
- Network Throughput
- CPU Resources

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 with UltraSPARC® III systems.

Disk Throughput

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/Os 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).

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. 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.

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

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, the Siroe Company used 100 Mbps Ethernet for client access and 1 GB Ethernet for the backbone.
- To ensure that sufficient capacity exists for future expansion, don't use theoretical maximum values when configuring your network.
- Separate traffic flows on different network partitions to reduce collisions and to optimize bandwidth use.

CPU Resources

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

Developing 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.

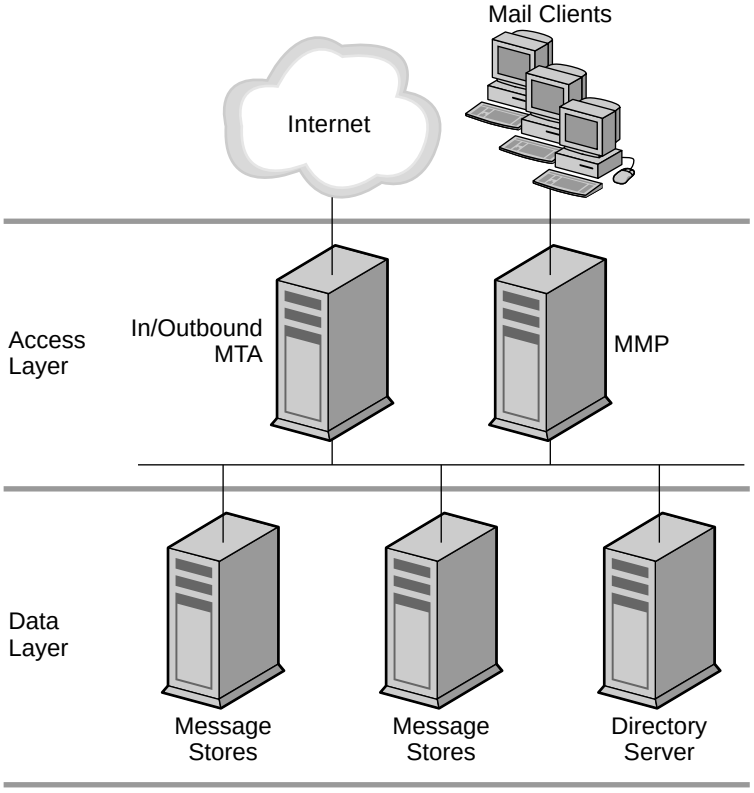
The following sections point out sizing considerations when you deploy two-tiered and one-tiered architectures.

NOTE For detailed information on planning your architecture, see Chapter 3, "Developing a Messaging Architecture."

Two-tier Architecture

A two-tier 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. The following figure shows a simplified two-tiered architecture.

Figure 6-1 Simplified 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 Load Simulator" on page 100.
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.

Sun Professional Services is often asked to specify a recommendation for the maximum number of users on a message store. Note that such a recommendation cannot be given without understanding:

- Usage patterns (as described in "Using a Load Simulator" on page 100).
- 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 MTA Routers

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 routers is to determine the maximum number of messages that can be relayed per second.

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

1. From the raw performance of the inbound router, 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 routers 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.

In addition, sufficient disk capacity for network problems or non-functioning remote MTAs must be calculated for transient messages.

➤ **To Size Your Multiplexing Services**

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

NOTE These instructions assume that you are installing the MEM and the MMP on the same machine.

In addition, you must:

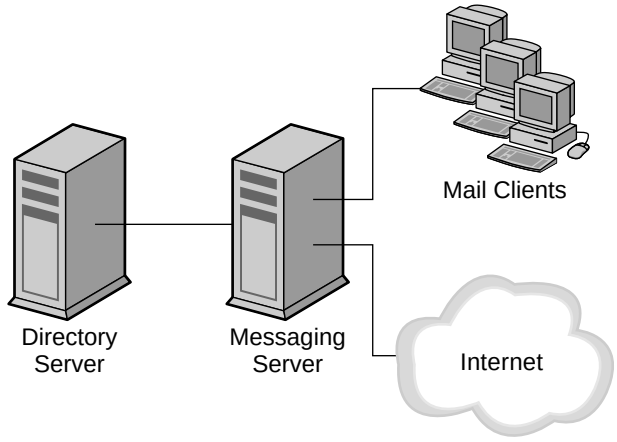
1. Add CPU or a hardware accelerator for SSL on MMP and MEM if appropriate.
2. Add memory to the machine if the MEM is being configured on it.
3. Add more disks for an SMTP proxy for the MMP.
4. Account for Denial of Service.
5. Add capacity for load balancing and redundancy, if appropriate.

   Like 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.

One-tier Architecture

In a one-tier 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 6-2 on page 107 shows a one-tier architecture.

Figure 6-2 Simplified One-tiered Architecture



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

➤ To Size a One-tier Architecture

1. Size your message stores like you size message stores in a "Two-tier Architecture" on page 104.
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 one-tier or two-tier architectures, contact your Professional Services representative.

Chapter 7

Understanding Messaging Server Schema and Provisioning Options

This chapter 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.

This chapter contains the following sections:

Understanding Messaging Schema Choices

This section describes the two schema options that are available and supported with Messaging Server, and how to decide which to use.

NOTE See the `commdirmig` command in the *Sun Java System Communications Services Schema Migration Guide* for information on how to migrate from Sun Java System LDAP Schema version 1 to Sun Java System LDAP Schema version 2.

Support for installation and provisioning of Schema 1 will be deprecated and removed from future releases. However, customers with their own provisioning tools may continue to use LDAP Schema 1.

Deciding Which Schema to Use

Choosing the schema that's right for your installation depends on your provisioning needs:

- Are you integrating Messaging Server with other Java Enterprise System component products, such as Sun Java System Portal Server or Sun Java System Identity Server, which provides single sign-on capabilities?

  If you answer Yes, then you must use Schema 2.

- Are you installing Messaging Server for the first time or are you upgrading from an older version?

  If you are installing Messaging Server for the first time, use Schema 2.

  If you are upgrading from an older version of Messaging Server, you can either use Schema 1 or Schema 2.

- Do you plan to use an interface for your provisioning needs? If so, will it be a graphical interface or a command-line interface?

  If you need to use a graphical user interface, or if you need your end users to be able to make modifications to their profiles through a graphical user interface, you should use Schema 1. Note that this option is not available for new installations of Messaging Server. It is only available for existing Messaging Server 5.x installations where Messaging Server 6 is now being installed.

  If you are planning on using a command-line interface, you can use Schema 1 (for existing Messaging Server installations) or Schema 2 (for new or for existing Messaging Server installations).

LDAP Schema 1

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 previous Messaging Server 5.x versions.

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.

Only sites that have installed previous versions of Messaging Server should use Schema 1.

NOTE Migrating to Schema 2 is imperative if you plan to install Messaging Server with other Sun Java System products in the future.

LDAP Schema 1 Supported Provisioning Tools

Schema 1 supports Sun™ ONE Delegated Administrator for Messaging as well as LDAP provisioning tools. For more information, see "Understanding Messaging Server Provisioning Tools" on page 112.

Schema 2 (Native Mode)

Schema 2 is a newly defined 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 LDAP directory server. 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 on your machine that are dependent on a two-tree provisioning model. You should also use this mode if you want to install other products in the Java Enterprise System product suite.

If you have an existing Messaging Server 5.x installation that uses Schema 1, and you want to integrate Messaging Server with other Java Enterprise Server products, you should migrate your directory to Schema 2 after you upgrade to Messaging Server 6. Refer to the *Sun Java System Communications Services Schema Migration Guide* for information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2.

NOTE Schema 2 Native Mode is the recommended provisioning model for all Sun Java System products in the Java Enterprise System product suite.

LDAP Schema 2 Supported Provisioning Tools

Schema 2 supports Sun Java System Communications Services User Management Utility. For more information, see "Understanding Messaging Server Provisioning Tools" on page 112.

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. Schema 2 compatibility mode also assumes that you have installed Identity Server prior to installing Messaging Server.

Use Schema 2 Compatibility if you have existing applications that require Schema 1, but you also need functionality that requires Schema 2, for example, Identity Server, single sign-on, and so forth.

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 the *Sun Java System Communications Services Schema Migration Guide* for more information.

Understanding Messaging Server Provisioning Tools

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.

In addition to the questions asked in "Deciding Which Schema to Use" on page 110, you should use Table 7-1 on page 114 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:

- Sun ONE Delegated Administrator for Messaging

Sun ONE Delegated Administrator for Messaging

Sun ONE Delegated Administrator for Messaging provides both a command-line and a graphical user interface to provision users and groups. Delegated Administrator uses Sun LDAP Schema 1, which is the Messaging Server 5.x version of provisioning definitions.

LDAP Provisioning Tools

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.

User Management Utility

Sun Java System Identity Server uses Schema 2. Because the Sun Java System component products in the Java Enterprise System product suite use Schema 2, use the Communications Services 6 User Management Utility. This should particularly be the case if you are using more than one Java Enterprise System product, or if you are performing a brand new installation of Messaging Server.

NOTE Even though you install Identity Server, there is no graphical user interface compatibility with Messaging Server. Therefore, to provision users and groups with an interface, you can only use the user management utility.

See the *Sun Java System Communications Services User Management Utility Administration Guide* for installation details.

Comparing Your Provisioning Tool Options

Table 7-1 on page 114 shows the various supported schema, provisioning tools, provisioning limitations, and recommended documentation for additional information.

Table 7-1 Messaging Server Provisioning Mechanisms

Supported Provisioning Tool	Provisioning Tool Functionality	Provisioning Tool Limitations	For Further Information
Sun ONE Delegated Administrator for Messaging Graphical User Interface Uses: Schema 1	Provides a graphical user interface for administrators to manage users, groups, domains, and mailing lists. End users can manage vacation messages and sieve filters.	• Only available to existing Messaging Server 5.x customers who are now upgrading to Messaging Server 6. • Can only be used with Sun ONE Web Server 6.0 (which is only available with the Messaging Server 5.2 bundle). It cannot be used with Sun ONE Web Server 6.1. • Incompatible with Sun Schema 2 and with other Java Enterprise System products. • Unable to use mail filters through Sun Java System Messenger Express. Must use filters through Delegated Administrator. • Must use auto reply channel which is only available in Messaging Server 5.2 product.	Read the Sun ONE Delegated Administrator for Messaging 1.3 documentation. Describes how to install and administer the Sun ONE Delegated Administrator interface.
Sun ONE Delegated Administrator for Messaging Command-line Interface Uses: Schema 1	Provides a command-line interface for administrators to manage users, groups, domains, and mailing lists.	• Incompatible with Sun Schema 2 and with other Java Enterprise System products.	Read the Sun ONE Delegated Administrator for Messaging 1.3 documentation. Provides syntax and usage for Sun ONE Delegated Administrator command-line utilities.

Table 7-1 Messaging Server Provisioning Mechanisms *(Continued)*

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 Sun Schema 2 and with other Java Enterprise System products.	Read the *Sun ONE Messaging Server 5.2 Provisioning Guide* and *Sun ONE Messaging and Collaboration Schema Reference Manual*. Describes the Sun LDAP Schema 1 provisioning model. In addition, these guides explain how to use LDAP provisioning tools and the usage of specific attributes and object classes.
Sun Java System Console Uses: Schema 1	Though provisioning functionality is included in the Sun Java System Console, it is not recommended for provisioning Messaging users and groups. Instead, use Sun Java System Console to administer server configuration such as quotas, log files, and other related Message Store items.	• Incompatible with Sun Schema 2 and with other Java Enterprise System products. • Not recommended as a provisioning tool in that the Console is unable to properly add and modify users and groups.	Read the *Sun Java System Messaging Server Administration Guide* and corresponding Sun Java System Console Online Help.
User Management Utility Uses: Schema 2	Provides a command-line interface for administrators to manage users, groups, domains, and mailing lists. Compatible with other Java Enterprise System products.	• Not backwardly compatible with Sun Schema 1. • No GUI provisioning tool to use with Sun Java System Identity Server • Sun Java System Identity Server must be installed to enable this command-line interface.	Read the *Sun Java System Communications Services User Management Utility Administration Guide*. Provides syntax and usage for the command-line utility.

Chapter 8

Planning Your Anti-Spam and Anti-Virus Strategy

Messaging Server provides many tools for dealing with unsolicited bulk email (UBE, or “spam”) and viruses. This chapter describes the various tools and strategies available for your use.

This chapter contains the following sections:

Anti-Spam and Anti-Virus Tools Overview

As more computers are connected to the Internet, and the ease of doing business online increases, the frequency of security incidents, including spam and viruses, continues to rise. You should plan your Messaging Server deployment to deal with these problems.

Mail traffic passing into, through, and out of Messaging Server can be separated into distinct channels according to various criteria. This criteria includes source and destination email addresses as well as source IP address or subnet. You can apply different processing characteristics to these different mail flows, or channels. Consequently, you can use different access controls, mail filters, processing priorities, and tools in different ways and combinations on these channels. For example, you can process mail originating from within your domain differently from mail originating from outside your deployment.

In addition to channel-based message flow classification, another useful classification is mailing list traffic. Traffic for a given mailing list can come into Messaging Server through a number of different channels and go back out through a number of different channels. When using mailing lists, you can find it helpful to think in terms of the list itself and not in terms of channels. Messaging Server recognizes this and enables many of the channel-specific spam fighting tools to also be applied in a mailing-list specific fashion.

The following summarizes the anti-spam and anti-virus tools you can use with Messaging Server:

- **Access Control.** Rejects mail from known spam sources and enables control over who can send or receive email within the organization
- **Mailbox Filtering.** Enables users to manage their own spam filters through a Web interface, controlling the nature of mail delivered to their mailboxes
- **Address Verification.** Refuses mail with invalid originator addresses
- **Real-time Blackhole List.** Refuses mail from recognized spam sources as identified by the Mail Abuse Protection System‘s Real-time Blackhole List (MAPS RBL), a responsibly managed, dynamically updated list of known spam sources
- **Relay Blocking.** Prevents abusers from using a mail system as a relay to send their spam to tens of thousands of recipients
- **Authentication Service.** Enables password authentication in an SMTP server with the Simple Authentication and Security Layer (SASL) protocol
- **Sidelining.** Silently sidelines or even deletes potential spam messages
- **Comprehensive Tracing.** Use reliable mechanisms for identifying a message’s source
- **Conversion Channel.** Integrates with third-party anti-virus or anti-spam products.

You can use these tools individually or together. No one tool by itself will block all spam. However, taken together, these tools provide an effective means of combatting unauthorized use of your mail system. The following sections provide more details on these tools.

Access Controls

Messaging Server has a general purpose mechanism that you can use to reject mail in accordance with a variety of criteria. This criteria includes the message source or destination email addresses, as well as source IP address. For example, you can use this mechanism to refuse mail from specific senders or entire domains (such as mail from `spam@public.com`). Should you have large lists of screening information, you can extend your lists with a database that stores the access criteria. While not UBE-related, this same access control mechanism is also suitable for maintaining a database of internal users who are or are not allowed to send mail out certain channels. For example, you can restrict on a per-user basis who can or cannot send or receive Internet mail.

See"Access Controls" on page 132 and the *Sun Java System Messaging Server Adminstration Guide* for more information.

Mailbox Filtering

Messaging Server provides a mail filters on a per-user, per-channel, and system-wide basis. Per-user channels can be managed from any web browser in Messenger Express. Using these filters, users can control what mail messages are delivered to their mailbox. For example, a user tired of "make money fast" UBE can specify that any message with such a subject be rejected. Mail filtering in Messaging Server is based on the Sieve filtering language (RFCs 3028 and 3685) being developed by the Internet Engineering Task Force (IETF)

See "Using Mailbox Filters" on page 134 and the *Sun Java System Messaging Server Adminstration Guide* for more information.

You can also implement content-based filtering of virus scanning through the use of third-party content filtering software, such as Brightmail and SpamAssassin. See "Anti-Spam and Anti-Virus Considerations" on page 122 for more information.

Address Verification

UBE messages often use invalid originator addresses. The Messaging Server SMTP server can take advantage of this by reflecting messages with invalid originator addresses. If the originator address does not correspond to a valid host name, as determined by a query to the DNS server, the message can be rejected. Note that a potential performance penalty can be incurred with such use of the DNS.

You enable address verification on a per-channel basis with the `mailfromdnsverify` channel keyword described in the *Sun Java System Messaging Server Adminstration Guide*.

Real-time Blackhole List

The Mail Abuse Protection System's Real-time Blackhole List (MAPS RBL) is a dynamically updated list of known UBE sources identified by source IP address. The Messaging Server SMTP server supports use of the MAPS RBL and can reject mail coming from sources identified by the MAPS RBL as originators of UBE. The MAPS RBL is a free service provided through the Internet DNS.

For more information, see:

`http://mail-abuse.org/rbl`

Use of the RBL by the Messaging Server SMTP server is enabled with the `ENABLE_RBL` option of the MTA Dispatcher. See the *Sun Java System Messaging Server Adminstration Guide* for more information.

Relay Blocking

A comprehensive UBE strategy should include both ways to prevent users from receiving UBE (access controls, mailbox filtering, address verification, RBL) as well as preventing users from unauthorized relay of mail from your system to other systems. This second method is called relay blocking. In its simplest form, relay blocking is achieved by enabling local users and systems to relay mail while rejecting relay attempts from non-local systems. Using IP addresses as the differentiator easily and securely makes this differentiation between local versus non-local. By default, Messaging Server enables relay blocking upon installation. See "Configuring Anti-Relaying with Mapping Tables" on page 134 and the *Sun Java System Messaging Server Adminstration Guide* for more information.

Authentication Services

The Messaging Server SMTP server implements the Simple Authentication and Security Layer (SASL, RFC2222) protocol. SASL can be used with POP and IMAP clients to provide password-based access to your SMTP server. A typical usage for SASL is to permit mail relaying for external authenticated users. This solves the common problem posed by local users who use ISPs from home or while traveling. Such users, when connecting to your mail system, will have non-local IP addresses.

Any relay blocking that takes into account only the source IP address will not permit these users to relay mail. This difficulty is overcome through the use of SASL, which enables these users to authenticate themselves. Once authenticated, the users are permitted to relay mail.

Sidelining

The access control mechanisms discussed previously can also defer the processing of suspect messages for later, manual inspection. Or, rather than sideline, the mechanisms can change the destination address, thus routing the suspect mail to a specific mailbox or simply deleting it silently. This tactic is useful when UBE is being received from a known, fixed origin and outright rejection will only cause the abuser to change the point of origin. Similar features are available for Messaging Server mailing lists. Great care should exercised when silently deleting mail to ensure that valid senders are not affected.

See the *Sun Java System Messaging Server Adminstration Guide* for more information.

Comprehensive Tracing

Messaging Server's SMTP server discovers and records crucial origination information about every incoming mail message, including, for example, source IP address and the corresponding host name. All discovered information is recorded in the message's trace fields (for example, the Received: header line) as well as in log files, if they are so configured. Availability of such reliable information is crucial in determining the source of UBE, which often has forged headers. Sites can use their own preferred reporting tools to access this information, which is stored as plain text.

Conversion Channel

The conversion channel is a very general purpose interface where you can invoke a script or another program to perform arbitrary body part processing of an email message. The conversion program hands off each MIME body part (not the entire message) to the program or script and can replace the body part with the output of the program or script. Conversion channels can be used to convert one file format to another (for example, text to PostScript), to convert one language to another, perform content filtering for company sensitive information, scan for viruses and replace them with something else.

Integration with Third-Party Products

Content-filtering software from third-party suppliers can be hooked in to your deployment through Messaging Server's conversion channel. Channel keywords are used to enable mail filtering using anti-spam and anti-virus products, such as Brightmail or SpamAssassin. You can configure the MTA to filter for all messages or only those going from or to certain channels, or to set the granularity at a per-user level. A user can opt in for spam or virus filtering or both. (SpamAssassin only filters for spam.)

An extensive Sieve support enables great flexibility to set the disposition of the message determined to be spam or virus. You can take the default action of discarding the virus and spam, or filing the spam into a special folder. But using Sieve, you can forward a copy of the message to some special account, add a custom header, or use the spamtest sieve extension to take different action based on a rating returned by SpamAssassin.

Anti-Spam and Anti-Virus Considerations

This section describes issues to keep in mind when planning your deployment to use anti-spam or anti-virus technologies.

Architecture Issues with Anti-Spam and Anti-Virus Deployments

The Messaging Server MTA can reside on the same system as the mail filtering system, such as Brightmail or SpamAssassin, or you can use separate systems. One of the advantages of separating the MTA from the mail filtering servers is that you can add more processing power for the filtering simply by adding more hardware and cloning the servers. While the system is capable and not overloaded, you can have the mail filtering server software collocated with the MTA.

In general, consider deploying a "farm" of Brightmail severs that the MTAs utilize to filter. You can configure MTAs to use a list of Brightmail server names, which essentially the MTAs will load balance on. (This load balancing functionality is provided by the Brightmail SDK.) The advantage of having the Brightmail server farm is that when you need more processing power, you can simply add more Brightmail servers.

Mail filtering products tend to be CPU-intensive. Creating an architecture that separates the MTA and the mail filtering products onto their own machines provides for better overall performance of the messaging deployment.

NOTE Because mail filtering servers tend to be CPU-intensive nature, you could end up with an architecture consisting of more mail filtering systems than the MTA hosts they are filtering for.

In larger deployments, consider also creating inbound and outbound mail filtering pools of servers that are associated with the respective inbound and outbound MTA pools. You can also create a “swing” pool that can be utilized as either an inbound or outbound pool, in response to need in either area.

As with the rest of the deployment, you need to monitor the mail filtering tier. A threshold of 50 percent CPU utilization is a good rule of thumb to follow. Once this threshold has been met, you need to consider adding more capacity to the mail filtering tier.

Implementing an RBL

In general, implementing an RBL provides the most immediate benefit to reducing spam traffic. A good RBL implemented by your MTAs immediately reduces spam by a minimum of 10 percent. In some cases, this number could approach 50 percent.

You can use your RBL and Brightmail together. If Brightmail takes care of 95 out of 100 emails for a certain IP address within some amount of time you should add that IP address to your RBL. You can adjust the RBLs for Brightmail’s false positives when you do your Brightmail analysis. That makes the RBL much more proactive in handling a specific wave of spam.

Common Anti-Spam and Anti-Virus Deployment Scenarios

This section describes common deployment scenarios for Brightmail and SpamAssassin. See the *Sun Java System Messaging Server Adminstration Guide* for more information:

http://docs.sun.com/doc/817-6267

Using Brightmail

There are several common deployment scenarios for Brightmail:

- Processing incoming messages to the local message store (`ims-ms` channel)
- Processing messages going out to the internet (`tcp-local` channel)
- Processing messages coming in from the internet (`tcp-local` channel)
- Processing messages going to a specific domain (`per-domain` option)
- Processing messages going to specific users (`per-user` option)
- Setting up Brightmail processing as a Class-of-Service Option

If Brightmail implements both spam and virus checking, MTA message throughput can be reduced by as much 50 percent. To keep up with MTA throughput, you may need two Brightmail servers for each MTA.

Using SpamAssassin

Messaging Server supports the use of SpamAssassin, a freeware mail filter used to identify spam. SpamAssassin consists of a library written in Perl and a set of applications and utilities that can be used to integrate SpamAssassin into messaging systems.

SpamAssassin calculates a score for every message. Scores are calculated by performing a series of tests on message header and body information. Each test either succeeds or fails, and the score is adjusted accordingly. Scores are real numbers and may be positive or negative. Scores that exceed a certain threshold (typically 5.0) are considered to be spam.

SpamAssassin is highly configurable. Tests can be added or removed at any time and the scores of existing tests can be adjusted. This is all done through various configuration files. Further information on SpamAssassin can be found on the SpamAssassin Web site:

http://www.spamassassin.org

The same mechanism used for calling out to the Brightmail spam and virus scanning library can be used to connect to the SpamAssassin `spamd` server.

Developing an Anti-Spam and Anti-Virus Site Policy

When developing a policy for preventing spam and relaying, strike a balance between providing safety from spam and providing a site where emails are delivered in a timely fashion. The best policy is therefore to initially provide a core set of measures that do not take up too much processing time but trap the majority of spam. You can then define this core set of measures after stress testing the final architecture. Start with the initial measures below. Once you have deployed your system, monitor trapped and non-trapped spam to fine tune the system and replace or add new functions if required.

Use the following set of measures as a starting point for your site's anti-spam and anti-virus policy:

- Anti-relay should be provided by the `ORIG_SEND_ACCESS` settings. This is structured to enable only subscribers and partnership users access to deliver externally bound SMTP mail.
- Use authentication services to validate roaming users. These users verify their identity before being allowed to route externally bound SMTP mail.
- Implement subject line checking for common spam phrases using the system-wide mailbox filters.
- Set a maximum number of recipients using the `holdlimit` keyword. This will have the effect of sidelining potential spam traffic. The initial value could be set at 50 recipients and should be monitored over a period of time to determine whether a higher or lower value is required.
- Set up dummy accounts that are then manually used by the postmasters to encourage spam to these specific accounts to identify new spam sites.

- A message in which a virus has been detected should not be returned to the original sender and should not be forwarded to the intended recipient. There is no value in this because most viruses generate their own mail with forged sender addresses. It has become very rare that such infected messages will have any useful content.
- Send infected messages to an engine that harvests and catalogues information about the virus. You can then use such information to create threat reports for your system administrators about new virus and worm outbreaks.

Chapter 9

Designing a Secure Messaging Server

This chapter provides an overview of security methods, describes common security threats, and outlines the steps in analyzing your security needs.

This chapter contains the following sections:

Creating a Security Strategy

Creating a security strategy is one of the most important steps in planning your deployment. Your strategy should meet your organization's security needs and provide a secure messaging environment without being overbearing to your users.

In addition, your security strategy needs to be simple enough to administer. A complex security strategy can lead to mistakes that prevent users from accessing their mail, or it can allow users and unauthorized intruders to modify or retrieve information that you don't want them to access.

RFC 2196, the *Site Security Handbook*, lists five steps to developing a security strategy:

1. Identify what you are trying to protect.

   For example, your list might include hardware, software, data, people, documentation, network infrastructure, or your organization's reputation.

2. Determine what you are trying to protect it from.

   For example: unauthorized users, spammers, or denial of service attacks.

3. Estimate how likely threats are to your system.

   If you are a large Service Provider, your chances of security threats could be greater than a small organization. In addition, the nature of your organization could provoke security threats.

4. Implement measures that will protect your assets in a cost-effective manner.

   For example, the extra overhead in setting up an SSL connection can put a performance burden on your Messaging deployment. In designing your security strategy, you need to balance security needs against server capacity.

5. Continuously review your strategy and make improvements each time a weakness is found.

   Conduct regular audits to verify the efficiency of your overall security policy. You can do this by examining log files and information recorded by the SNMP agents. For more information on SNMP, refer to the *Sun Java System Messaging Server Administration Guide*.

Your security strategy should also plan for:

- Physical Security
- Server Security
- Network Security

- Messaging Security

Physical Security

Limit physical access to important parts of your infrastructure. For example, place physical limits on routers, servers, wiring closets, server rooms, or data centers to prevent theft, tampering, or other misuse. Network and server security become a moot point if any unauthorized person can walk into your server room an unplug your routers.

Server Security

Limiting access to important operating system accounts and data is also part of any security strategy. Protection is achieved through the authentication and access control mechanisms available in the operating system.

In addition, you should install the most recent operating environment security patches and set up procedures to update the patches once every few months and in response to security alerts from the vendor.

Network Security

Limiting access to your network is an important part of your security strategy. Normally, overall access to networks is limited through the use of firewalls. However, email must be made available outside your site. SMTP is one such service.

To secure your network, you should:

- Turn off all operating system-provided services that listen on ports that you do not use.
- Replace `telnet` with `sshd`, if possible.
- Place your messaging server behind a packet filter, which drops external packets with an internal source IP address. A packet filter forbids all connections from the outside except for those ports that you explicitly specify.

Messaging Security

Messaging Server offers the following sets of security features:

- Protecting Messaging Components in Your Deployment

  With this set of options, you can secure your MTA relays, message stores, Messenger Express mail clients, and multiplexing services. In addition, you'll learn about third-party spam filter options.

- Planning User Authentication

  These options enable you to determine how your users will authenticate to your mail servers, preventing unauthorized users from gaining access to your system.

- Planning Message Encryption Strategies

  Using this set of options, you can perform user authentication and protect the message itself by using authenticated SMTP and certificates for digital signatures, encryption, and Secure Sockets Layer (SSL).

The remainder of this chapter describes these security methods for protecting your Messaging system.

Protecting Messaging Components in Your Deployment

This section describes how to secure components in your Messaging deployment:

NOTE With each component, you should use the `chroot` function to limit the number of available commands on each machine.

Protecting MTA Relays

Secure MTA relays to protect processing resources and server availability. When messages are relayed from unauthorized users or large quantities of spam are delivered, response time is reduced, disk space is used up, and processing resources, which are reserved for end users, are consumed. Not only does spam waste server resources, it is also a nuisance for your end users.

NOTE Not only must you protect your deployment from external unauthorized users, but you might also have to protect your system from internal users as well.

The following table describes the most common threats to MTA relays.

Table 9-1 Common MTA Relay Security Threats

Threat	Description
UBE (Unsolicited bulk email) or spam	Refers to the practice of sending electronic junk mail to millions of users.
Unauthorized Relaying	Uses another company's SMTP server to relay your email. Spammers often use this technique to cover their tracks. End-users might send complaints back to the sending relay, not to the spammer.
Mail Bombs	Characterized by abusers who repeatedly send an identical message to a particular address. The goal is to exceed mailbox quotas with the message.
Email Spoofing	Creates email that appears to have originated from one source when it actually was sent from another source.
Denial of Service Attacks	Prevents legitimate users of a service from using that service. For example, an attacker attempts to flood a network, thereby preventing legitimate network traffic.

This section on MTA relays describes security options you can use in your deployment:

- Access Controls
- Conversion Channels and Third Party Filtering Tools
- RBL Checking
- Client Access Filters
- Monitoring Your Security Strategy

Access Controls

You can use access controls to reject messages from (or to) certain users at a system level. In addition, you can institute more complex restrictions of message traffic between certain users. Also, you might allow users to set up filters on their own incoming messages (including rejecting messages based on contents of the message headers).

If you want to control access with envelope-level controls, use mapping tables to filter mail. If you want to control access with header-based controls, or if users wish to implement their own personalized controls, use the more general mailbox filters approach with server-side rules.

Mapping Table Overview

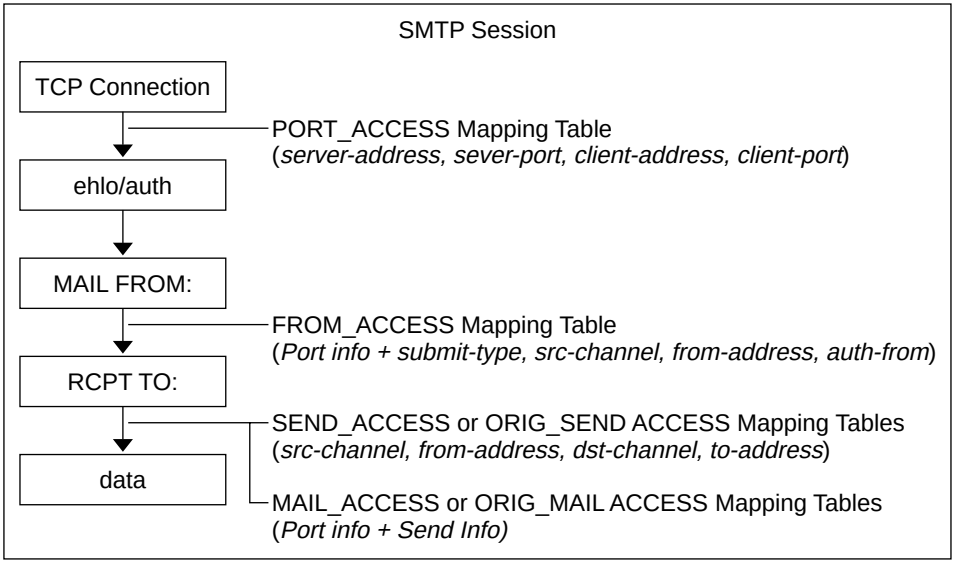
You can control access to your mail services by configuring certain mapping tables. The following table describes these mapping tables, which enable you to control who can or cannot send mail, receive mail, or both. See the *Sun Java System Messaging Server Adminstration Guide* for more information.

Table 9-2 Access Control Mapping Tables

Mapping Table	Description
SEND_ACCESS	Used to block incoming connections based on envelope From: address, envelope To: address, source and destination channels. The To: address is checked after rewriting, alias expansion, and so on, have been performed.
ORIG_SEND_ACCESS	Used to block incoming connections based on envelope From: address, envelope To: address, source and destination channels. The To: address is checked after rewriting but before alias expansion.
MAIL_ACCESS	Used to block incoming connections based on combined information found in SEND_ACCESS and PORT_ACCESS tables: that is, the channel and address information found in SEND_ACCESS combined with the IP address and port number information found in PORT_ACCESS.
ORIG_MAIL_ACCESS	Used to block incoming connections based on combined information found in ORIG_SEND_ACCESS and PORT_ACCESS tables: that is, the channel and address information found in ORIG_SEND_ACCESS combined with the IP address and port number information found in PORT_ACCESS.
FROM_ACCESS	Used to filter mail based on envelope From: addresses. Use this table if the To: address is irrelevant.
PORT_ACCESS	Used to block incoming connections based on IP number.

The following figure illustrates where mapping tables are activated in the mail acceptance process.

Figure 9-1 Mapping Tables and the Mail Acceptance Process



For all the network ports controlled by the MTA service dispatcher, a `PORT_ACCESS` rejection response, if warranted, takes place at the initial connection from a remote host. A `FROM_ACCESS` rejection occurs in response to the `MAIL FROM:` command, before the sending side can send the recipient information or the message data. A `SEND_ACCESS` or `MAIL_ACCESS` rejection occurs in response to a `RCPT TO:` command, before the sending side gets to send the message data. If an SMTP message is rejected, your Messaging Server never accepts or sees the message data, thus minimizing the overhead of performing such rejections. If multiple access control mapping tables exist, Messaging Server checks them all.

NOTE If the message is accepted, it can still be filtered by way of conversion channels and user defined filters.

Configuring Anti-Relaying with Mapping Tables

You can also use access control mappings to prevent people from relaying SMTP mail through your Messaging Server system. For example, someone might try to use your mail system to relay junk mail to thousands of mailboxes on your system or on other systems.

By default, Messaging Server prevents all SMTP relaying activity, including relaying by local POP and IMAP mail clients. If IMAP and POP clients do not authenticate by using SMTP AUTH, as described in "Enabling Authenticated SMTP" on page 143, and attempt to submit messages to external addresses via Messaging Server's SMTP server, their submission attempts are rejected. Thus, you will likely want to modify your configuration so that it recognizes your own internal systems and subnets from which relaying should always be accepted.

➤ **To Prevent Relaying From Outside Hosts**

To prevent hosts that reside outside your domain from relaying to other hosts outside your domain:

1. Split incoming mail into different channels. For example:
   - IP addresses within your domain go to the `tcp_internal` channel.
   - Authenticated sessions go to the `tcp_auth` channel.
   - All other mail is sent to the `tcp_local` channel.
2. Recognize and allow mail from your POP and IMAP clients by using an `INTERNAL_IP` mapping table, fully explained in the chapter on Mail Filtering and Access Control in the *Sun Java System Messaging Server Administration Guide*.

Using Mailbox Filters

A filter consists of one or more conditional actions to apply to a message. Messaging Server filters are stored on the server and evaluated by the server. They are sometimes called server-side rules (SSR).

You can create channel-level filters and MTA-wide filters to prevent the delivery of unwanted mail. You can also create filter templates and make them available to end users by using Messenger Express. End users use the templates to build personal mailbox filters to prevent delivery of unwanted mail message to their mailboxes. The server applies filters in the following priority. See the *Sun Java System Messaging Server Administration Guide* for more information.

1. Per-user filters

   Per-user filters apply to messages destined for a particular user's mailbox. You can create filter templates and make them available to end users by using the Messenger Express client. End users use the templates to build personal server filters to manipulate the delivery of mail messages to their mailboxes. The filers reject unwanted messages, redirect mail, filter messages into mailbox folders, and so on.

   If a personal mailbox filter explicitly accepts or rejects a message, then filter processing for that message finishes.

   A filter template generalizes a Sieve script by replacing "hard-coded" elements of the Sieve script with prompts and input fields. A Java servlet is used to parse the sieve templates and generate the user interface in the browser. When an end user supplies values in the input fields, the servlet takes those values and saves them in a sieve script in the user's directory profile entry. The prompts and input fields are presented to the end user through the Messenger Express interface.

   If the recipient user had no mailbox filter, or if the user's mailbox filter did not explicitly apply to the message in question, Messaging Server next applies the channel-level filter.

2. Channel-level filter

   Channel-level filters apply to each message enqueued to a channel. A typical use for this type of filter is to block messages going through a specific channel.

   To create a channel-level filter, you must write the filter using SIEVE. See the Mail Filtering and Access Control chapter in the *Sun Java System Messaging Server Administration Guide* for specific instructions on creating filters with SIEVE.

   If the channel-level filter explicitly accepts or rejects a message, then filter processing for that message finishes. Otherwise, Messaging Server next applies the MTA-wide filter, if there is one.

3. MTA-wide filter

   MTA-wide filters apply to all messages enqueued to the MTA. A typical use for this type of filter is to block unsolicited bulk email or other unwanted messages regardless of the messages' destinations.

   To create an MTA-wide filter, you must write the filter using SIEVE. See the Mail Filtering and Access Control chapter in the *Sun Java System Messaging Server Administration Guide* for specific instructions on creating filters with SIEVE.

   By default, each user has no mailbox filter. When a user accesses Messenger Express interface to create one or more filters, then their filters are stored in the LDAP Directory.

Conversion Channels and Third Party Filtering Tools

The conversion channel performs body-part-by-body-part conversions on messages through the MTA. This processing can be done by any site-supplied programs or command procedures. The conversion channel can do such things such as convert text or images from one format to another, scan for viruses, translate languages, and so forth. Various message types of the MTA traffic are selected for conversion, and specific processes and programs can be specified for each type of message body part. If you are looking to use the conversion channel with a virus scanning program, you can either disinfect, hold, or reject messages. A special conversion channel configuration is consulted to choose an appropriate conversion for each body part. For more information, see the Using Pre-defined Channels chapter in the *Sun Java System Messaging Server Administration Guide.*

NOTE Using specialized processing like a conversion channel puts additional load on your system. Be sure to account for it when you plan your sizing strategy.

With the conversion channel, you can use third-party anti-spam and anti-virus software solutions. You can also use the MTA API to create a channel to invoke a remote scanning engine. For more information on the MTA API, see the *Sun Java System Messaging Server Developer's Reference.*

In general, it is best that these third-party solutions are shielded from external sites and are only used on back-end or intermediate relays.

The Brightmail solution consists of the Brightmail server and real-time anti-spam and anti-virus (for service providers only) rule updates that are downloaded to your messaging servers. When the Brightmail Logistics and Operations Center (BLOC) receives spam from email probes, operators immediately create appropriate anti-spam rules. These rules are then downloaded to Brightmail customer machines. Similarly, the Symantec Security Response real-time virus rules are also sent from Brightmail. These rules are used by customer's Brightmail servers to catch spam and viruses.

Messaging Server also supports the use of SpamAssassin, a freeware mail filter used to identify spam. SpamAssassin calculates a score for every message. Scores are calculated by performing a series of tests on message header and body information. Each test either succeeds or fails, and the score is adjusted accordingly. Scores are real numbers and may be positive or negative. Scores that exceed a certain threshold are considered to be spam.

For more information on configuring Brightmail and SpamAssassin for Messaging Server, see the *Sun Java System Messaging Server Administration Guide*.

RBL Checking

The Mail Abuse Protection System's Real-time Blackhole List (MAPS RBL) is a list of host and networks that are known to be friendly or neutral to abusers who use these hosts and networks to either originate or relay spam, or to provide spam support services.

You can configure your MTA relays to compare incoming connections against the MAPS RBL. You can also use DNS-based databases used to determine incoming SMTP connections that might send unsolicited bulk mail.

For more information, see the Mail Filtering and Access Control chapter in the *Sun Java System Messaging Server Administration Guide*.

Client Access Filters

Messaging Server supports sophisticated access control on a service-by-service basis for POP, IMAP, and HTTP. The Messaging Server access-control facility is a program that listens at the same port as the TCP daemon it serves. The access-control facility uses access filters to verify client identity, and it gives the client access to the daemon if the client passes the filtering process.

If you are managing messaging services for a large enterprise or for a service provider, these capabilities can help you to exclude spammers and DNS spoofers from your system and improve the general security of your network.

As part of its processing, the Messaging Server TCP client access-control system performs (when necessary) the following analyses of the socket end-point addresses:

- Reverse DNS lookups of both end points (to perform name-based access control)
- Forward DNS lookups of both end points (to detect DNS spoofing)
- `Identd` callback (to check that the user on the client end is known to the client host)

The system compares this information against access-control statements called *filters* to decide whether to grant or deny access. For each service, separate sets of Allow filters and Deny filters control access. Allow filters explicitly grant access; Deny filters explicitly forbid access.

When a client requests access to a service, the access-control system compares the client's address or name information to each of that service's filters—in order—using these criteria:

1. The search stops at the first match. Because Allow filters are processed before Deny filters, Allow filters take precedence.
2. Access is granted if the client information matches an Allow filter for that service.
3. Access is denied if the client information matches a Deny filter for that service.
4. If no match with any Allow or Deny filter occurs, access is granted. The exception is the case where there are Allow filters but no Deny filters, in which case lack of a match means that access is denied.

The filter syntax described here is flexible enough that you should be able to implement many different kinds of access-control policies in a simple and straightforward manner. You can use both Allow filters and Deny filters in any combination, even though you can probably implement most policies by using almost exclusively Allows or almost exclusively Denies.

Client access filters are particularly helpful if troublesome domains are a known quantity. While UBE filters must store and process every spam message, client access filters free Messaging Server from having to process any spammed messages. Because client access filters block mail from entire domains, this feature should be used with caution.

Note the following limitations to client access filters:

- An SMTP client is required to log in before relaying a message.

- Client access filters do not scale well for large deployments.

For more information on client access filters, see the Configuring Security and Access Control chapter in the *Sun Java System Messaging Server Administration Guide*.

Monitoring Your Security Strategy

Monitoring your server is an important part of your security strategy. To identify attacks on your system, monitor message queue size, CPU utilization, disk availability, and network utilization. Unusual growth in the message queue size or reduced server response time may identify some of these attacks on MTA relays. Also, investigate unusual system load patterns and unusual connections. Review logs on a daily basis for any unusual activity.

Protecting the Message Store

The most important data in a messaging server is the user's mail in the message store. Note that the mail messages are stored as individual files, which are not encrypted. Consequently, physical access and `root` access to the message store must be protected.

To secure the Message Store, restrict access to the machine where the store is installed. You can enable CRAM-MD5 or DIGEST-MD5 passwords instead of using unencrypted, plaintext passwords. For more information on passwords, see "Planning User Authentication" on page 141.

Not only should you create password authentication to the store machine, you might also use tools like VPN access, `ssh`, or `pam`, which lists valid users that are allowed to login to the machine.

In addition, a two-tier architecture is recommended over a one-tier architecture. Because the Message Store performs the most disk intensive work of any components in a messaging system, do not have filtering, virus scanning, and other disk-intensive security processes on the same machine. In a two-tier architecture, you don't have to run UBE filters, anti-relay, and client access filters on the same machine as the message store, which can add load to your system. Instead, the MTA relays handle that processing. In addition, user access to the store is limited to through an MMP or an MEM (Messenger Express Multiplexor) in a two-tier deployment, potentially adding an extra security layer to the message store.

If you deploy a one-tier architecture, be sure to account for the additional security processing and load (like SSL and virus scanning) that you will need. For more information, see Chapter 6, "Planning Your Sizing Strategy."

For additional Message Store security processing, set disk quotas per user to limit disk usage. Also, use administrator alarms if free space thresholds are fast approaching their limits. Like the MTA, be sure to monitor the server state, disk space, and service response times. For more information, see the Managing Your Message Store chapter in the *Sun Java System Messaging Server Administration Guide*.

Protecting MMP and Messenger Express Multiplexor (MEM)

Because the MMP serves as a proxy for the Message Store, it needs to protect access to end user data and guard against unauthorized access. User IDs and passwords provide basic authentication capabilities. In addition, you can use client access filters to limit user login to specific domains or IP address ranges. SMTP Authentication, or SMTP AUTH (RFC 2554) is the preferred method of providing SMTP relay server security. SMTP AUTH allows only authenticated users to send mail through the MTA. For more information, see "Enabling Authenticated SMTP" on page 143.

Locate the MMP on a different machine (or under a different userID) in front of your POP or IMAP services. You can have front-end machines with just MMP and MTA relays, and then have a physically secure network between those front-end machines, the mail stores, and the LDAP servers.

Special security considerations need to be given to Messenger Express access to the Message Store when your users are logging in from the Internet. In general, you want to make sure that the stores are separated from the outside world by a firewall. In addition, you might consider adding a Messenger Express Multiplexor (MEM), a specialized server that acts as a single point of connection to the HTTP access service. Like the MMP, the MEM supports both unencrypted and encrypted (SSL) communication with mail clients. The MEM also has to protect access to end user data and guard against unauthorized access.

Regular monitoring of log files can protect against unauthorized access.

Planning User Authentication

User authentication enables your users to log in through their mail clients to retrieve their mail messages. Methods for user authentication include:

- Plain Text and Encrypted Password Login
- Authentication with Simple Authentication and Security Layer (SASL)
- Enabling Authenticated SMTP
- Certificate-based Authentication with Secure Sockets Layer (SSL)

Plain Text and Encrypted Password Login

User IDs and passwords are stored in your LDAP directory. Password security criteria, such as minimum length, are determined by directory policy requirements. Password security criteria is not part of Messaging Server administration. To understand directory server password policies, see the *Sun Java System Directory Server Deployment Planning Guide*. An administrator can set a messaging configuration parameter to determine if plain passwords are allowed or if passwords must be encrypted. For more information, see the `service.`*xxx*`.plaintextminciper` (where *xxx* is `http`, `pop`, or `imap`) parameter in the *Sun Java System Messaging Server Administration Reference*.

Both plain text and encrypted password login can be used with POP, IMAP, and Messenger Express user access protocols.

Authentication with Simple Authentication and Security Layer (SASL)

SASL (RFC 2222) provides additional authentication mechanisms for POP, IMAP, and SMTP user access protocols. Messaging Server has SASL support for the user access protocols listed in the following table:

Table 9-3 SASL Authentication User Access Protocols Support Matrix

	Plain	Login	CRAM-MD5	Digest-MD5	Certificate	APOP
SMTP AUTH	Yes	Yes	Yes	Yes	-	-
POP	Yes	-	Yes	Yes	-	Yes
IMAP	Yes	-	Yes	Yes	-	-

Table 9-3 SASL Authentication User Access Protocols Support Matrix *(Continued)*

	Plain	Login	CRAM-MD5	Digest-MD5	Certificate	APOP
HTTP (Messenger Express)	Yes	-	-	-	Yes	-

NOTES

- When using CRAM-MD5, passwords must be stored in plain text format in the LDAP directory server.
- Digest-MD5 is not yet supported in the MMP, but it is supported if you choose not to use the MMP.
- When using POP, passwords must be stored in plain text format in the LDAP directory server.

If you use SASL, user name and passwords are not encrypted unless SSL is used for the session. (For more information on SSL, see "Encryption with SSL" on page 145). The SASL mechanisms, CRAM-MD5, DIGEST-MD5, and LOGIN, encode authentication information, but can be easily decoded if captured. Despite this limitation, SASL is useful because it can be combined with SMTP AUTH (described in "Enabling Authenticated SMTP" on page 143) to allow only authenticated users to send mail to relay mail through your system. For example, legitimate users can authenticate to the SMTP server, and the SMTP server can then be configured to switch to a different channel. In this way, the message from an authenticated session can come from a different TCP channel than a user that did not authenticate. A message from a user in your internal network can also be switched to differentiate it from a message coming from other sources just based on the IP address of the incoming connection.

For more information on SASL, see the Configuring Security and Access Control chapter in the *Sun Java System Messaging Server Administration Guide*.

Enabling Authenticated SMTP

Authenticated SMTP (also referred to as SMTP AUTH) is an extension to the SMTP protocol. Authenticated SMTP allows clients to authenticate to the server. The authentication accompanies the message. The primary use of authenticated SMTP is to enable local users who are not in their office to submit mail without creating an open relay that others could abuse. The AUTH command is used by the client to authenticate to the server.

Authenticated SMTP provides security in sending messages with the SMTP protocol. To use authenticated SMTP, you do not need to deploy a certificate-based infrastructure. (Certificates authentication is described in "Certificate-based Authentication with Secure Sockets Layer (SSL).")

With authenticated SMTP, the client can indicate an authentication mechanism to the server, perform an authentication protocol exchange, and optionally negotiate a security layer for subsequent protocol interactions. For example, if supported by your mail client, authenticated SMTP can require your end users to enter a password before they can send messages.

If you require SMTP AUTH for mail submission, you turn on appropriate logging, so any mail abuse can be traced.

For more information on authenticated SMTP, see the MTA chapters of the *Sun Java System Messaging Server Administration Guide*.

Certificate-based Authentication with Secure Sockets Layer (SSL)

Messaging Server uses the SSL protocol for encrypted communications and for certificate-based authentication of clients and servers. This section describes certificate-based SSL authentication. For information on SSL Encryptions, see "Encryption with SSL" on page 145.

SSL is based on the concepts of public-key cryptography. Although TLS (Transport Layer Security) is functionally a superset of SSL, the names are used interchangeably.

At a high-level, a server which supports SSL needs to have a certificate, a public key, a private key, certificate, key, and security databases. This helps assure message authentication, privacy, and integrity.

Table 9-4 on page 144 describes the SSL authentication support with each client access protocol.

Table 9-4 SSL Authentication Support Matrix

	SSL with MMP	SSL with MMP on Alternate Port	SSL	SSL on Alternate Port
SMTP	Yes	Yes	Yes	Yes
POP	-	Yes	-	Yes
IMAP	Yes	Yes	-	Yes
Messenger Express (HTTP)	Yes (through Messenger Express Multiplexor)	Yes (through Messenger Express Multiplexor)	Yes	Yes

The SMTP, POP, IMAP, and HTTP protocols provide a way for the client and server to start communication without SSL, and then switch to it by using an equivalent “start TLS” command. The SMTP, POP, IMAP, and HTTP servers can also be configured to only use SSL on an alternate port, if the client and server do not implement “start TLS.”

To authenticate with SSL, the mail client establishes an SSL session with the server and submits the user’s certificate to the server. The server then evaluates if the submitted certificate is genuine. If the certificate is validated, the user is considered authenticated.

If you use SSL for authentication, you need to obtain a server certificate for your Messaging Server. The certificate identifies your server to clients and to other servers. Your server can have more than one server certificate with which it identifies itself. Your server can also have any number of certificates of trusted Certification Authorities (CAs) that it uses for client authentication.

For more information on SSL, see the Security and Access Control chapter in the *Sun Java System Messaging Server Administration Guide*.

Planning Message Encryption Strategies

This section describes encryption and privacy solutions. The following topics are covered:

- Encryption with SSL
- Signed and Encrypted S/MIME

Encryption with SSL

SSL functions as a protocol layer beneath the application layers of IMAP, HTTP, and SMTP. If transmission of messages between a Messaging Server and its clients and between the servers and other servers is encrypted, there is little chance for eavesdropping on the communications. If connecting clients and servers are authenticated, there is little chance for intruders to spoof them.

End-to-end encryption of message transmission may require the use of SSL with SMTP, IMAP, and HTTP protocols.

NOTE The extra performance overhead in setting up an SSL connection can put a burden on the server. In designing your messaging installation and in analyzing performance, you need to balance security needs against server capacity.

If you use SSL for encryption, you can improve server performance by installing a hardware encryption accelerator. An encryption accelerator typically consists of a hardware board, installed permanently in your server machine, and a software driver.

The SSL connection process between client and server using HTTP/SSL (HTTPS) is as follows:

1. The client initiates contact using HTTPS. The client specifies which secret-key algorithms it can use.
2. The server sends its certificate for authentication and specifies which secret-key algorithm should be used. It will specify the strongest algorithm which it has in common with the client. If there is no match (for example, client is 40 bit only, server requires 128 bits), the connection will be refused.
3. If the server has been configured to require client authentication, it will ask the client for its certificate at this point.

4. The client checks the validity of the server certificate to make sure that it has:
   - Not expired
   - A known signed Certification Authority
   - A valid signature
   - A host name in the certificate matches the same of the server in the HTTPS request

SSL Ciphers

A cipher is the algorithm used to encrypt and decrypt data in the encryption process. Some ciphers are stronger than others, meaning that a message a cipher has scrambled is more difficult for an unauthorized person to unscramble.

A cipher operates on data by applying a key to the data. Generally, the longer the key the cipher users during encryption, the more difficult it is to decrypt the data without the proper decryption key.

When a client initiates an SSL connection with Messaging Server, the client lets the server known what ciphers and key lengths it prefers to use for encryption. In any encrypted communication, both parties must use the same ciphers. Because there are a number of cipher-and-key combinations in common use, a server should be flexible in its support for encryption. For more information on ciphers, see the chapter on Configuring Security and Access Control in the *Sun Java System Messaging Server Administration Guide*.

Signed and Encrypted S/MIME

Messages that are signed and encrypted are referred to as Secure/Multipurpose Internet Mail Extensions (S/MIME) messages. S/MIME is a means of securing client to client communication.

With S/MIME, senders can encrypt messages prior to sending them. These recipients can store the encrypted messages after receipt, decrypting them only to read them. Using S/MIME requires no special Messaging Server configuration or tasks. It is strictly a client action. Unlike SSL, it provides end-to-end encryption. See your client documentation for information on setting up S/MIME.

Understanding Security Misconceptions

This section describes common messaging misconceptions that are counterproductive to the security needs of your deployment.

- **Hiding Product Names and Versions**

  At best, hiding product names and versions hinders casual attackers. At worst, it gives a false sense of security that might cause your administrators to become less diligent about tracking real security problems.

  In fact, removing product information and version numbers makes it more difficult for the vendor support organization to validate software problems as being that of their software or that of other software.

  Hackers have little reason to be selective, particularly if there is a known vulnerability in SMTP servers, where they may attempt to access any SMTP server.

  A determined individual can use other protocol behaviors to determine the vendor name and version regardless of any attempt to hide it.

- **Hiding Names of Internal Machines**

  Hiding internal IP addresses and machine names will make it more difficult to:

  - Trace abuse or spam
  - Diagnose mail system configuration errors
  - Diagnose DNS configuration errors

  A determined attacker will have no problem discovering the machine names and IP addresses of machines once they find a way to compromise a network.

- **Turning Off EHLO on the SMTP Server**

  Without EHLO you also lose:

  - NOTARY
  - TLS negotiation
  - Preemptive controls on message sizes

With EHLO, the remote SMTP client determines if you have a limit and stops trying to send a message that exceeds the limit as soon as it sees this response. But, if you have to use HELO (because EHLO is turned off), the sending SMTP server sends the entire message data, then finds out that the message has been rejected because the message size exceeds the limits. Consequently, you are left with wasted processing cycles and disk space.

- **Network Address Translation (NAT)**

  If you use NAT to provide a type of firewall, you do not have an end-to-end connection between your systems. Instead, you have a third node which stands in the middle. This NAT system acts as a middleman, causing a potential security hole.

Other Security Resources

For more information on designing a secure Messaging deployment, review the Computer Emergency Response Team (CERT) Coordination Center site:

http://www.cert.org

Chapter 10

Planning for Service Availability

This chapter helps you determine the level of service availability that is right for your deployment. The level of service availability is related to the hardware you choose as well as your software infrastructure and maintenance practices. This chapter discusses several choices, their value, and their costs.

This chapter contains the following sections:

Automatic System Reconfiguration (ASR) Overview

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 Sun SPARC systems support very different levels of ASR. Some systems support no ASR to 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.

Understanding High Availability Models

You can use a variety of high availability models for Messaging Server. Three of the more common models are:

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

The following subsections describe each of these models in more detail.

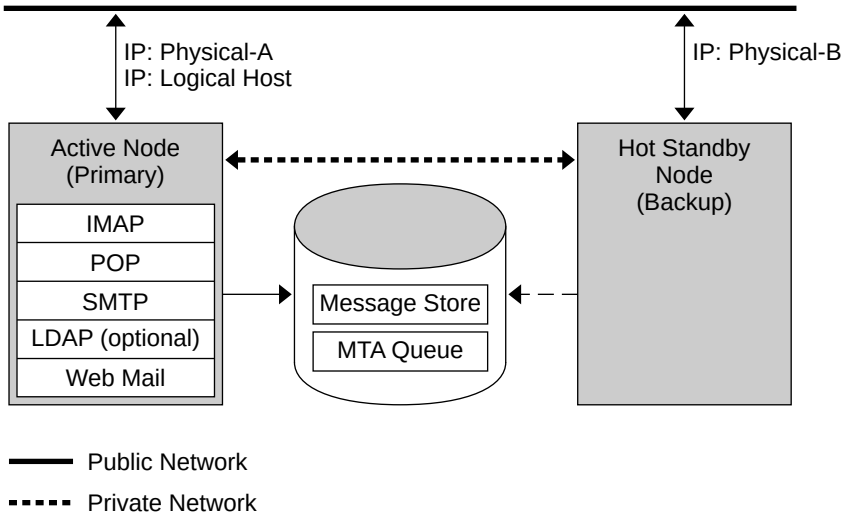
NOTE Different HA products potentially do or do not support different models. Refer to the appropriate product HA documentation to determine which models are supported.

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 Message Transfer Agent (MTA) queues reside on this shared volume. The following figure shows the asymmetric model.

Figure 10-1 Asymmetric High Availability Model



The preceding figure 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`.

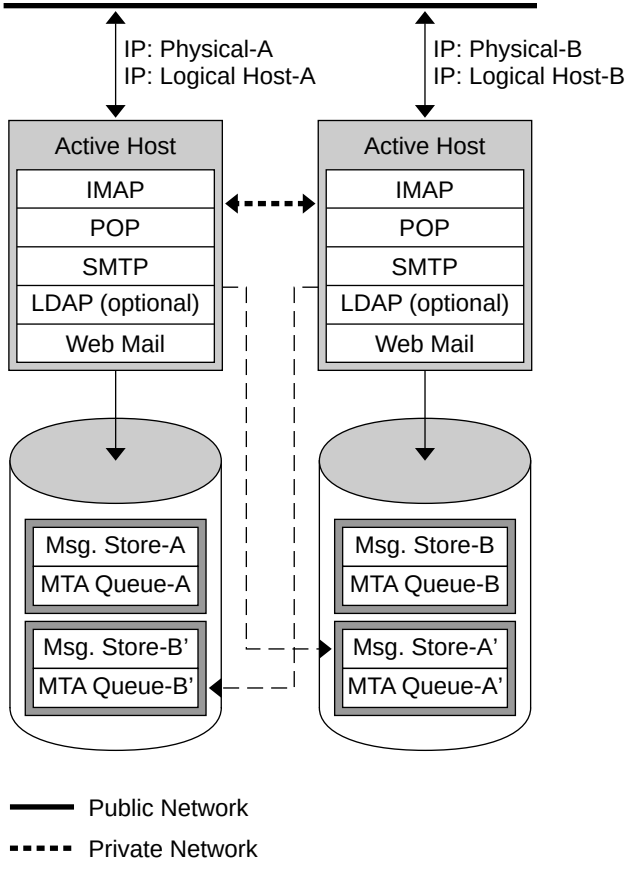
The advantage of this model is that the backup node is dedicated and completely reserved for the primary node. Additionally, there is no resource contention on the backup node when a failover occurs. However, this model also means that the backup node stays idle most of the time and this resource is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

The following figure 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 Availability 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.

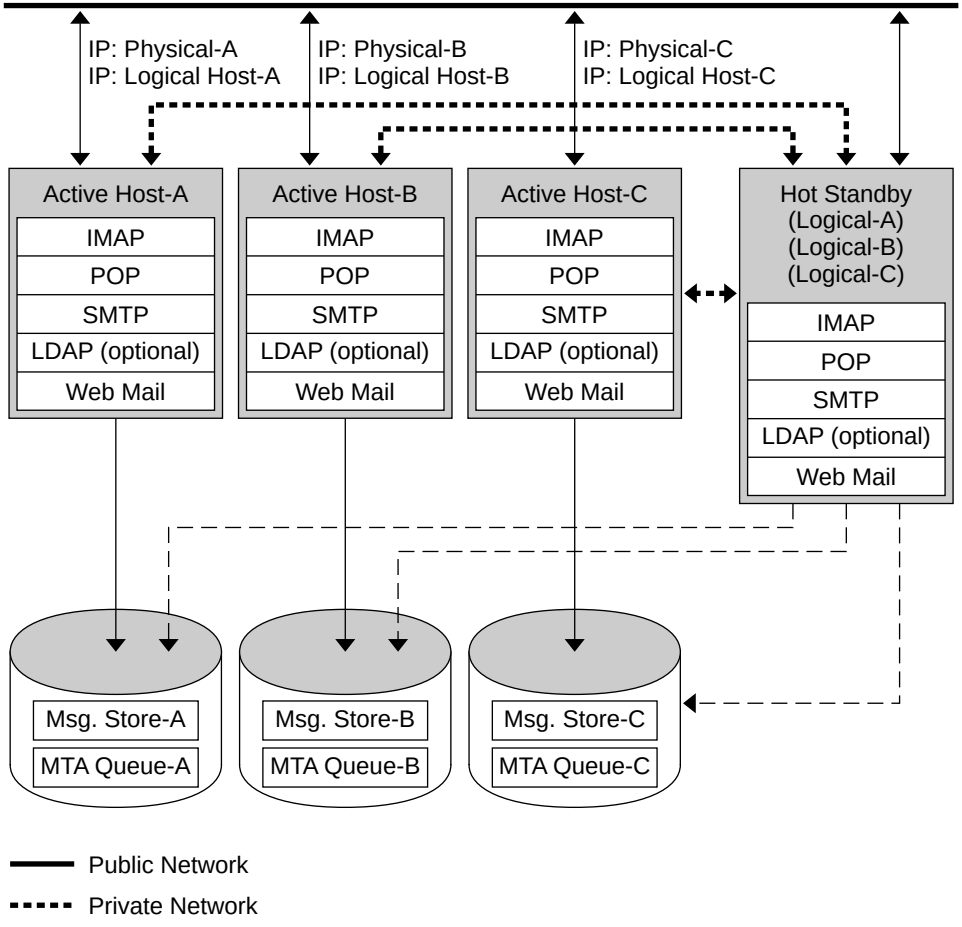
To configure a symmetric model, you need to install shared binaries on your shared disk. Note that doing so might prevent you from performing rolling upgrades, a feature that enables you to update your system during Messaging Server patch releases. (This feature is planned for future releases.)

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 on page 154 illustrates the basic N + 1 high availability model.

Figure 10-3 N + 1 High Availability 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.

Choosing a High Availability Model

The following table 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 10-1 High Availability Model Advantages and Disadvantages

Model	Advantages	Disadvantages	Recommended User
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

System Down Time Calculations

The following table 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 10-2 System Down Time Calculations

Model	Server Down Time Probability
Single server (no high availability)	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 + 1 day of storage down)/365 = 0.27%
N + 1 Asymmetric	Pr(down) = (5 hours of system down + 1 day of storage down)/(365xN) = 0.33%/N

Locating Product Reference Information

For more information on high availability models supported by Messaging Server, see the following product documentation:

- **Sun Cluster**
  - *Sun Cluster Concepts Guide for Solaris OS*
  - *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*
  - *Sun Cluster Overview for Solaris OS*
  - *Sun Cluster System Administration Guide for Solaris OS*
- **Veritas Cluster Server**
  - *Veritas Cluster Server User's 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 disappears 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 (hopefully without experiencing DoS failures) 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 `imrestore` process; and the file system restore is started following which services are brought up.
- **More complicated, more expensive solution.** Both Veritas and Sun 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. Instead, a Hitachi Data Systems (HDS) array performs this function. Hitachi arrays have the ability to do this data copy between arrays with little to no impact on the store servers. The HDS arrays are large, so the base cost of obtaining this solution is higher than a few Sun StorEdge™ T3 or 3000 arrays. Also, the cost per megabyte is higher even if the HDS is fully utilized. If you need a large quantity of storage, the savings in server hardware due to allowing the HDS do the copy work can balance the additional cost of the storage, at least to some degree.

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 put a number on. 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 that user's 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 Sun Cluster for local HA and Veritas software for remote site failover. Sun 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.

Chapter 11

Pre-Installation Considerations and Procedures

This chapter describes considerations you need to think about, and procedures you need to perform, before installing Messaging Server. See the *Sun Java Enterprise System 2004Q2 Installation Guide* for instructions on running the Java Enterprise System installer.

This chapter contains the following sections:

Installation Considerations

This section describes installation considerations that help you prepare to install Messaging Server.

- **Resource Contention.** To avoid resource contention between the servers, considering installing Directory Server on a different host than where you install Messaging Server.
- **Installation Privileges.** You must install Messaging Server logged in as `superuser`.
- **Messaging Server Base Directory.** The Messaging Server is installed into a directory referred to as *msg_svr_base* (for example, `/opt/SUNWmsgsr`). This directory provides a known file location structure (file directory path).

- **Upgrading Servers.** If you do not install other component products (Web Server, Directory Server, Identity Server, and Administration Server) on the Messaging Server host, you do not have to upgrade those components and Messaging Server should continue to operate without problem. If other component products are installed on the same machine, then they must be upgraded along with Messaging Server.
- **Port Number Conflicts.** If certain products are installed on the same machine, you will encounter port number conflicts. The following table shows potential port number conflicts.

Table 11-1 Potential Port Number Conflicts

Conflicting Port Number	Component	Component
143	IMAP Server	MMP IMAP Proxy
110	POP3 Server	MMP POP3 Proxy
993	IMAP over SSL	MMP IMAP Proxy with SSL
80	Identity Server (Web Server port)	Messenger Express

If possible, install products with conflicting port numbers on separate hosts. If you are unable to do so, then you will need to change the port number of one of the conflicting products. To change port numbers, use the `configutil` utility (see the *Sun Java System Messaging Server Administration Reference* for instructions).

The following example uses the `service.http.port configutil` parameter to change the Messenger Express HTTP port number to `8080`.

```
configutil -o service.http.port -v 8080
```

Installation Worksheets

When installing Messaging Server, use the following installation worksheets to record and assist you with the installation process. You can reuse these installation worksheets for multiple installations of Messaging Server, uninstallation, or for Messaging Server upgrades.

TIP Record all the port numbers you specify during the installation, along with the specific component using that port number.

The following worksheets are included:

- Directory Server Installation Worksheet
- Administration Server Initial Runtime Configuration Worksheet

Directory Server Installation Worksheet

You either installed Directory Server through the Java Enterprise System installer, or have a previous Directory Server installation. Record your Directory Server installation and configuration parameters in the following table. You will need these parameters when you install and configure Administration Server and Messaging Server, as well as your initial Messaging Server runtime configuration. For additional help, see the *Sun Java System Messaging Server Administration Guide*.

Table 11-2 Directory Server Installation Parameters

Parameter	Description	Example	Used in	Your Answer
Directory Installation Root	A directory on the Directory Server host dedicated to holding the server program, configuration, maintenance, and information files.	`/var/opt/mps/serverroot`	`comm_dssetup.pl` Perl script	

Table 11-2 Directory Server Installation Parameters *(Continued)*

Parameter	Description	Example	Used in	Your Answer
Host	The fully qualified domain name. The fully qualified domain name consists of two parts: the host name and the domain name.	`svr1.west.sesta.com`	Administration Server Configuration	
LDAP Directory Port Number	The default for an LDAP directory server is 389.	`389`	Administration Server Configuration and Messaging Server Configuration	
Administrator ID and Password	Administrator in charge or responsible for configuration information. Password for the Administrator	`Admin` `PaSsWoRd`	Administration Server Configuration	
User and Group Tree Suffix	The distinguished name of the LDAP entry at the top of the directory tree, below which user and group data is stored.	`o=usergroup`	`comm_dssetup.pl` Perl script	
Directory Manager DN and Password	The privileged directory administrator, comparable to `superuser` in UNIX. Typically, this administrator is responsible for user and group data. Password for the Directory Manager.	`cn=Directory Manager` `pAsSwOrD`	`comm_dssetup.pl` Perl script and Messaging Server Configuration	
Administration Domain	A region of administrative control.	`System Lab`	Administration Server Configuration	

Administration Server Initial Runtime Configuration Worksheet

When you run the Administration Server initial runtime configuration program through the Java Enterprise System installer, record your installation parameters in the following table. You will need some of these parameters for the Messaging Server initial runtime configuration. You might also refer to your "Directory Server Installation Worksheet" on page 163 to answer certain questions.

Table 11-3 Administration Server Initial Runtime Configuration Program Parameters

Parameter	Description	Example	Your Answer
Fully Qualified Domain Name	Fully qualified domain name for the host machine.	`svr1.west.sesta.com`	
Server Root Definition	Installation Root of the Administration Server dedicated to holding the server program, configuration, maintenance, and information files.	`/var/opt/mps/serverroot`	
UNIX System User	Certain privileges designated to system users to ensure they have appropriate permissions for the processes they are running. Always use `root`.	`root`	
UNIX System Group	The group to which certain UNIX System users belong. Always use `other`.	`other`	
Configuration Directory Server	Host and Port specified on Directory Server Installation Worksheet.	Host `svr1.west.sesta.com` Port `390`	
Configuration Directory Server Administrator and Password	Administrator ID specified on Directory Server Installation Worksheet. Password of Administrator ID	`Admin` PaSsWoRd	

Table 11-3 Administration Server Initial Runtime Configuration Program Parameters *(Continued)*

Parameter	Description	Example	Your Answer
Administration Domain	A region of administrative control. If you have installed Messaging Server and Directory Server on the same machine, then you should choose the same Administration Domain on Directory Server Installation Worksheet.	System Lab2	
Administrative Server Port	A unique port number dedicated to the Administration Server.	5555	

Choosing Which Messaging Server Components to Configure

When you install Messaging Server software, the Java Enterprise System installer installs all the Messaging Server packages. You then configure the appropriate Messaging Server component (MTA, Message Store, Messenger Express, MMP) on a Messaging host through the Messaging Server configuration program.

The following table shows which components you need to configure for each type of Messaging host.

Table 11-4 Which Messaging Server Components to Configure?

Type of Messaging Host Being Configured	Needs These Components Selected in the Configurator Program
MTA Relay	Message Transfer Agent
Message Store (back end)	Message Transfer Agent, Message Store, Messenger Express Note: You need to configure the store for an MEM proxy after configuration is done.
Messenger Express (front end only, no store or SMTP function)	Messenger Express, Messaging Multiplexor Note: If you are only configuring Messenger Express, you must also select the Message Store and the MTA, or at least be able to point to an existing MTA.

Table 11-4 Which Messaging Server Components to Configure? *(Continued)*

Type of Messaging Host Being Configured	Needs These Components Selected in the Configurator Program
Message Multiplexor (front end only, no store or SMTP function)	Messaging Multiplexor

NOTE Configuring the LMTP delivery mechanism requires configuration on both the relay machines and on the back end stores. See the *Sun Java System Messaging Server Administration Guide* for instructions on configuring LMTP.

Disabling the sendmail Daemon

Prior to installing Messaging Server, you should disable the `sendmail` daemon if it is running. The Dispatcher, under which the Messaging Server SMTP server runs, needs to bind to port 25. If the `sendmail` daemon is running (on port 25), the Dispatcher will not be able to bind to port 25.

➤ **To Disable the sendmail Daemon**

1. Change to the `/etc/init.d` directory.

```
cd /etc/init.d
```

2. Stop the `sendmail` daemon if it is running.

```
./sendmail stop
```

3. Modify `/etc/default/sendmail` by adding `MODE=""`.

   If the `sendmail` file does not exist, create the file and then add `MODE=""`.

   If a user accidentally runs `sendmail start`, or if a patch restarts `sendmail`, then adding this modification prevents `sendmail` from starting up in daemon mode.

Glossary

Refer to the Java Enterprise System Glossary (http://docs.sun.com/doc/816-6873) for a complete list of terms that are used in this guide.

Index

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W