Sun Java™ System

Communications Services 6 Deployment Planning Guide

2005Q1

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

Part No: 819-0063-10

Contents

List of Figures

List of Tables

Preface

The *Sun Java System Communications Services 6 2005Q1 Deployment Planning Guide* contains the information you need to deploy Sun Java™ System Communications Services 6 2005Q1. This guide helps you through the process of understanding Communications Services, evaluating and analyzing your site, and designing the kind of deployment architecture that meets your organization's needs.

This preface contains the following sections:

Who Should Use This Book

This guide is for individuals who are responsible for assessing and deploying Communications Services at your site, including:

- Evaluators
- Architects
- System administrators

Before You Read This Book

This guide assumes you are familiar with the following:

- How to design and install enterprise-level software products
- IMAP, POP, HTTP, SMTP, WCAP, and LDAP protocols
- Solaris™ Operating System (Solaris OS) system administration and networking

How This Book Is Organized

The first part of this book provides an overview of the entire Communications Services products and high-level deployment topics. Part II provides detailed information on deploying Sun Java™ System Messaging Server. Part III provides detailed information on deploying Sun Java™ System Calendar Server. Part IV provides detailed information on deploying Sun Java™ System Instant Messaging. Part V provides detailed information on deploying Sun Java™ System Communications Express. Part VI provides deployment examples. The following table summarizes the content of this book.

Table 1 How This Book Is Organized

Chapter	Description
Chapter 1, "Introduction to Deploying Communications Services"	Provides an overview of Communications Services.
Chapter 2, "Analyzing Your Communications Services Requirements"	Explains how to analyze your organization's business and technical requirements.

Table 1 How This Book Is Organized *(Continued)*

Chapter	Description
Chapter 3, “Understanding Product Requirements and Considerations”	Describes requirements and considerations that impact the design of your deployment.
Chapter 4, “Determining Your Network Infrastructure Needs”	Describes the components of your network infrastructure and how to plan your infrastructure layout.
Chapter 5, “Developing a Communications Services Logical Architecture”	Describes how to develop your Communications Services logical architecture.
Chapter 6, “Designing for Service Availability”	Discusses your service availability choices, their value, and their costs.
Chapter 7, “Designing for Security”	Provides an overview of security methods, describes common security threats, and outlines the steps in analyzing your security needs.
Chapter 8, “Understanding Schema and Provisioning Options”	Describes the schema and provisioning options for Communications Services.
Chapter 9, “Introduction to Messaging Server Software”	Provides an overview of the Messaging Server software.
Chapter 10, “Planning a Messaging Server Sizing Strategy”	Introduces the basics of sizing your Messaging Server deployment to enable you to obtain the right sizing data by which you can make deployment decisions.
Chapter 11, “Developing a Messaging Server Architecture”	Describes how to design the architecture of your Messaging Server deployment.
Chapter 12, “Designing a Messaging Server Topology”	Describes how to design your messaging topology, which is the physical and logical layout of a networked messaging system.
Chapter 13, “Planning Messaging Server Security”	Describes how to plan for and protect the various components of your Messaging Server deployment.
Chapter 14, “Planning a Messaging Server Anti-Spam and Anti-Virus Strategy”	Describes the various anti-spam and anti-virus tools and strategies available for your use.
Chapter 15, “Understanding Messaging Server Pre-Installation Considerations and Procedures”	Describes considerations you need to think about, and procedures you need to perform, before installing Messaging Server.
Chapter 16, “Introduction to Calendar Server Software”	Provides an overview of the Calendar Server software.
Chapter 17, “Developing a Calendar Server Architecture”	Describes basic architectures of your Calendar Server deployment.
Chapter 18, “Planning Calendar Server Security”	Describes how to plan for and protect the various components of your Calendar Server deployment.

Table 1 How This Book Is Organized *(Continued)*

Chapter	Description
Chapter 19, “Planning Calendar Server Services”	Describes the additional considerations of a Calendar Server deployment, with respect to Calendar Server services.
Chapter 20, “Understanding Calendar Server Pre-Installation Considerations”	Describes considerations you need to think about before installing Calendar Server.
Chapter 21, “Introduction to Instant Messaging Software”	Provides an overview of the Instant Messaging software.
Chapter 22, “Planning an Instant Messaging Sizing Strategy”	Introduces the basics of sizing your Instant Messaging deployment to enable you to obtain the right sizing data by which you can make deployment decisions.
Chapter 23, “Developing an Instant Messaging Architecture”	Describes a variety of Instant Messaging architectures.
Chapter 24, “Understanding Instant Messaging Pre-Installation Considerations”	Describes considerations you need to think about before installing Instant Messaging.
Chapter 25, “Introduction to Communications Express Software”	Provides an overview of the Communications Express software.
Chapter 26, “Developing a Communications Express Architecture”	Describes basic Communications Express architectures.
Chapter 27, “Understanding Communications Express Pre-Installation Considerations”	Describes considerations you need to think about before installing Communications Express.
Chapter 28, “Communications Services Deployment Examples”	Provides Communications Services deployment examples.
“Glossary”	Provides a link to the Java™ Enterprise System glossary.
“Index”	Provides an index to this book.

Conventions Used in This Book

The tables in this section describe the conventions used in this book.

Typographic Conventions

Table 2 on page 23 describes the typographic changes used in this book.

Table 2 Typographic Conventions

Typeface	Meaning	Examples
`AaBbCc123` (Monospace)	API and language elements, HTML tags, web site URLs, command names, file names, directory path names, onscreen computer output, sample code.	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`

`AaBbCc123` (Monospace bold)	What you type, when contrasted with onscreen computer output.	`%` **`su`**
		`Password:`

AaBbCc123 (Italic)	Book titles, new terms, words to be emphasized. A placeholder in a command or path name to be replaced with a real name or value.	Read Chapter 6 in the *User's Guide*. These are called *class* options. Do *not* save the file. The file is located in the *install-dir*/`bin` directory.

Symbols

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

Table 3 Symbol Conventions

Symbol	Description	Example	Meaning
`[ ]`	Contains optional command options.	`ls [-l]`	The `-l` option is not required.
`{ | }`	Contains a set of choices for a required command option.	`-d {y|n}`	The `-d` option requires that you use either the `y` argument or the `n` argument.
-	Joins simultaneous multiple keystrokes.	Control-A	Press the Control key while you press the A key.
+	Joins consecutive multiple keystrokes.	Ctrl+A+N	Press the Control key, release it, and then press the subsequent keys.

Table 3 Symbol Conventions *(Continued)*

Symbol	Description	Example	Meaning
>	Indicates menu item selection in a graphical user interface.	File > New > Templates	From the File menu, choose New. From the New submenu, choose Templates.

Default Paths and File Names

The following table describes the default path and file name used in this book.

Table 4 Default Path and File Name

Term	Description
product_base	Represents the base installation directory for Messaging Server. The Messaging Server 6 2005Q1 default base installation and product directory depends on your specific platform: Solaris systems: `/opt/SUNWmgsr` Linux systems: `/opt/sun/messaging`

Shell Prompts

The following table describes the shell prompts used in this book.

Table 5 Shell Prompts

Shell	Prompt
C shell on UNIX or Linux	*machine-name*%
C shell superuser on UNIX or Linux	*machine-name*#
Bourne shell and Korn shell on UNIX or Linux	`$`
Bourne shell and Korn shell superuser on UNIX or Linux	`#`
Windows command line	`C:\`

Related Documentation

The http://docs.sun.com^{SM} web site enables you to access Sun technical documentation online. You can browse the archive or search for a specific book title or subject.

Books in This Documentation Set

The following table summarizes the books included in the Communications Services core documentation set.

Other Server Documentation

Table 6 Books in This Documentation Set

Book Title	Description
Communications Services Schema Reference (http://docs.sun.com/doc/819-0113)	Serves as a reference for schema information for Sun Java System Communication Services products using LDAP, specifically Messaging Server and Calendar Server.
Communications Services Schema Migration Guide (http://docs.sun.com/doc/819-0112)	Describes how to migrate Sun Java™ System LDAP Directory data from LDAP Schema 1 to LDAP Schema 2 for Sun Java System Communications Services, specifically Messaging Server and Calendar Server.
Communications Services Delegated Administrator Guide (http://docs.sun.com/doc/819-0114)	Describes how to configure and administer Sun™ Java System Communications Services Delegated Administrator.

For other server documentation, go to the following:

- Access Manager documentation
  http://docs.sun.com/coll/AccessManager_05q1
- Calendar Server documentation
  http://docs.sun.com/coll/CalendarServer_05q1
- Communications Express documentation
  http://docs.sun.com/coll/MessagingServer_05q1
- Directory Server documentation
  http://docs.sun.com/coll/DirectoryServer_05q1

- Instant Messaging documentation
  `http://docs.sun.com/coll/InstantMessaging_05q1`
- Messaging Server documentation
  `http://docs.sun.com/coll/MessagingServer_05q1`

Accessing Sun Resources Online

For product downloads, professional services, patches and support, and additional developer information, go to the following:

- Download Center
  `http://wwws.sun.com/software/download/`
- Sun Java System Services
  `http://www.sun.com/service/sunps/sunone/index.html`
- Sun Enterprise Services, Solaris Software Patches, and Support
  `http://sunsolve.sun.com/`
- Developer Information
  `http://developers.sun.com/prodtech/index.html`

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in the product documentation, go to `http://www.sun.com/service/contacting`.

Related Third-Party Web Site References

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 or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions.

To share your comments, go to `http://docs.sun.com` and click Send Comments. In the online form, provide the document title and part number. 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 title of this book is *Sun Java System Communications Services 6 2005Q1 Deployment Planning Guide*, and the part number is 819-0063-10.

Part I

Deployment Planning Overview

Chapter 1

Introduction to Deploying Communications Services

This chapter provides an overview of Sun Java System Communications Services 6 2005Q1, the business reasoning behind deploying Communications Services, and the deployment process itself.

This chapter contains the following sections:

- Communications Services Overview
- How Communications Services Satisfy Business Needs
- Understanding the Deployment Process

Communications Services Overview

Sun Java System Communications Services 6 2005Q1 are a secure, cost-effective communications and collaborations offering. Communications Services address customer concerns about costs, capabilities, and the security of the traditional communications infrastructure by offering a secure, scalable, lower total cost of ownership alternative to other communications and collaboration solutions.

Communications Services provide the email, calendar, and instant messaging solutions necessary to meet both enterprise and ISP communications and collaboration needs. The products and services that form Communications Services provide a compelling response to common business requirements. All organizations need communications, and many are required to provide these services across large, diverse, and geographically distributed communities of users. Traditional communications solutions are costly, and not sufficient to meet today's requirements for scalability and security. Communications Services enable organizations to deploy solutions at a total cost of ownership they can afford.

In addition, Communications Services provide differentiated services and full-featured collaboration functionality that are required by a diverse audience. Finally, a Communications Services deployment meets your increased security needs when extending communications outside of a corporate firewall and to mobile users through multiple devices.

The Communications Services core solution consists of the following component products:

- Sun Java System Messaging Server 6 (formerly Sun™ ONE Messaging Server)
- Sun Java System Calendar Server 6 (formerly Sun™ ONE Calendar Server)
- Sun Java System Instant Messaging 7 (formerly Sun™ ONE Instant Messaging)

Additional features that enhance the Communications Services solution include:

- Sun Java™ System Communications Express 6
- Sun ONE™ Synchronization 1.1
- Sun Java™ System Connector for Microsoft Outlook 7

Taken as a whole, Communications Services provide a standards-based, integrated communications and collaboration suite of products for enterprise deployments of many thousands of users, and ISP deployments of hundreds of thousands of users. Communications Services deliver a robust and flexible platform meeting the diverse communications needs of all types of organizations. Communications Services are an optimal solution to connect remote offices, distributed workgroups, and global corporate locations.

About Messaging Server

Sun Java System Messaging Server 6 is a high-performance, highly secure messaging platform. Scaling from thousands to millions of users, Messaging Server is suitable for businesses interested in consolidating email servers and reducing total cost of ownership of communications infrastructure. Messaging Server provides extensive security features that help ensure the integrity of communications through user authentication, session encryption, and the appropriate content filtering to help prevent spam and viruses.

With Messaging Server, organizations can provide secure, reliable messaging services for entire communities of employees, partners, and customers.

Messaging Server currently supports two client user interfaces (UI):

- Messenger Express
- Communications Express

Going forward, no new features will be added to the Messenger Express user interface. It has been deprecated in favor of the Communications Express user interface. Sun Microsystems, Inc. will announce an end-of-life timeline for Messenger Express at a future date.

See Part II, "Deploying Messaging Server" for more information on Messaging Server concepts and other deployment aspects.

About Calendar Server

Sun Java System Calendar Server 6 facilitates team collaboration by enabling users to manage and coordinate appointments, events, tasks, and resources. With its intuitive, Web-based interface, Calendar Server enables end users to access their personal, public, or group calendars anytime, anywhere, from a Web browser. Deployments use Calendar Server, along with the Messaging Server and Instant Messaging, to offer users a comprehensive communications and collaborative environment.

Calendar Server currently supports two client user interfaces (UI):

- Calendar Express
- Communications Express

Calendar Express has been deprecated in favor of the new Communications Express user interface. Going forward, no new features will be added to the Calendar Express user interface. Sun Microsystems, Inc. will announce an end-of-life timeline for Calender Express at a future date.

See Part III, "Deploying Calendar Server" for more information on Calendar Server concepts and other deployment aspects.

About Instant Messaging

Sun Java System Instant Messaging 7 enables secure, real-time communication and collaboration. Instant Messaging combines presence awareness with instant messaging capabilities such as chat, conferences, alerts, news, polls, and file transfers to create a rich collaborative environment. These features enable one-to-one as well as group collaboration through either short-lived communications or persistent venues such as conference rooms or news channels. Instant Messaging, along with Calendar Server and Messaging Server, offers users a comprehensive communications and collaboration environment.

Instant Messaging ensures the integrity of communications through its multiple authentication mechanisms and secure SSL connections. Integration with Sun Java™ System Portal Server 6 and Sun Java™ System Access Manager 6 brings additional security features, services-based provisioning access policy, user management, and secure remote access. Furthermore, Instant Messaging supports the Extensible Messaging and Presence Protocol (XMPP). XMPP enables you to use a number of third-party clients, which aggregate contacts from the public networks. In one client, you can have contacts from AIM, Yahoo, MSN, Sun, and other XMPP-based servers.

See Part IV, "Deploying Instant Messaging" for more information on Instant Messaging concepts and deployment aspects.

About Communications Express

Sun Java System Communications Express 6 provides an integrated web-based communications and collaboration client. Communications Express is a common part of Messaging Server and Calendar Server, providing end users with a web interface to their calendar information and mail, as well as an address book.

See Part V, "Deploying Communications Express" for more information on Communications Express concepts and deployment aspects.

About Synchronization

Sun ONE Synchronization 1.1 is a software product that runs on a Windows personal computer and enables users to synchronize Calendar Server events and tasks with mobile devices and personal information managers (PIMs) such as Microsoft Outlook.

See the Sun ONE Synchronization documentation at the following location for more information:

`http://docs.sun.com/db/coll/S1_Sync_11`

About Connector for Microsoft Outlook

Sun Java System Connector for Microsoft Outlook 7 enables Outlook to be used as a desktop client with Messaging Server and Calendar Server.

Connector for Microsoft Outlook is an Outlook plug-in that you install on end-users' desktops. Connector for Microsoft Outlook queries Messaging Server for folder hierarchies and email messages. Connector for Microsoft Outlook then converts the information into Messaging API (MAPI) properties that Outlook can display. Similarly, Connector for Microsoft Outlook queries Calendar Server for events and tasks, which are then converted into MAPI properties. With this model, Connector for Microsoft Outlook builds an end-user Outlook view from two separate information sources: mail from Messaging Server and calendar information from Calendar Server.

Similarly, Connector for Microsoft Outlook uses Web Address Book Protocol (WABP) to query Address Book Server for contacts, which are then converted into MAPI properties. With this model, Connector for Microsoft Outlook builds an end-user Outlook view from three separate information sources: mail from Messaging Server, calendar information from Calendar Server, and contacts from Address Book Server.

See the Connector for Microsoft Outlook documentation at the following location for more information:

`http://docs.sun.com/db/coll/MessagingServer_05q1`

Communications Services Component Product Dependencies

Communications Services have dependencies on other Sun Java System component products that provide infrastructure services. These component products include Sun Java™ System Directory Server and, optionally, Sun Java System Access Manager. Additionally, Communication Services depend on a web server to serve HTML content and provide HTML connections. You can use Sun Java™ System Web Server (formerly Sun™ ONE Web Server) or Sun Java™ Application Server to fulfill this need.

Communications Services also depend on the existence of DNS. You need to have a functioning DNS server before you can install the Communications Services products.

See Chapter 3, "Understanding Product Requirements and Considerations" for more information on product dependencies.

How Communications Services Satisfy Business Needs

Organizations want to deploy services that simultaneously reduce cost and complexity while providing a robust set of features. The architecture of services must add requirements for security and scalability that enable users to have more than just a single means of accessing information critical to their daily work. Communications Services meet these needs through providing scalable messaging, calendaring, and instant messaging at a total cost of ownership businesses can afford.

Communications Services enable you to develop an architecture that incorporates ease of deployment and maintenance with a complete set of features and functionality. Most important, a Communications Services architecture builds security into each service element. These elements include the network infrastructure, operating environment, and the Communications Services component products themselves.

How Messaging Server Satisfies Business Needs

Messaging Server promotes superior reliability and productivity as well as reduced administrative and operational costs. Messaging Server uses committed transactions, which means that messages are not acknowledged as received until they are committed to disk. This reliability feature protects mail messages from loss and corruption. Additionally, the Message Store is built around a custom-designed database that employs a write-once data store and a two-level index to achieve excellent performance and data integrity.

How Calendar Server Satisfies Business Needs

Calendar Server provides one of the industry's most open, interoperable, and high-performance time and resource management solutions. Calendar Server provides the features you need at a lower total cost of ownership than alternative solutions. Through its flexible and extensible architecture, Calendar Server scales both vertically (by increasing the number of CPUs per system) and horizontally (by adding more servers to the network).

How Instant Messaging Satisfies Business Needs

Instant Messaging software is closely integrated with Java Enterprise System, helping you to shorten the project life cycle and deploy new services affordably. In addition, Instant Messaging works with Portal Server, Access Manager, Messaging Server, and Calendar Server. This integration provides users with a full-featured, secure, scalable communications and collaboration services platform from a single vendor. The well-documented Java APIs included in Instant Messaging provide open standards for ease of integration, as well as multiple platform support, platform extensibility, and customization of real-time communications and collaboration features. These features can thus be embedded in existing applications or become the basis of new applications. Also, XMPP interoperability provides a great advantage to those businesses seeking to extend their ability to practice real-time communication with their partners and customers, many of which might have separate instant messaging systems.

How Communications Express Satisfies Business Needs

Communications Express provides an integrated web-based communication and collaboration client that caters to the needs of Internet service providers, enterprises, and OEMs. Communications Express has an integrated user interface for calendar, mail, and address book and enables the access of one client module from another without re-authenticating user credentials. Communication between mail and calendar is established using Access Manager or Messaging Server single sign-on mechanism. Both calendar and mail applications share the same address book. All modules share the common user preferences specified in the Options tab of Communications Express.

Summary of Communications Services Benefits

The Communications Services components have been traditionally deployed in large-scale, carrier-class deployments. The same dependability required for the large-scale deployments can be used in the enterprise.

Table 1-1 on page 38 summarizes the benefits provided by Communications Services.

Table 1-1 How Communications Services Benefit Your Organization

Key Feature	Benefits
High performance and scalability	Enables efficient communications and improves quality of service for both enterprises and ISPs.
Extensive security features	Protects the integrity of communications and data and the privacy of employees, customers and partners, and enables compliance with industry regulations.
Virtual domain hosting and delegated administration	Messaging Server, Calendar Server, and Instant Messaging enable you to host messaging for several companies on one server, or corporate IT to host multiple departments within the organization, reducing number of servers needed, and lowering TCO.
Scalable, robust and extensible components	Enables deployment of unified communication services, bringing together telephone services with email notification, faxing, paging, and other technologies.
Extensible collaboration platform for scheduling events, and for managing tasks and resources	Calendar Server improves time and resource management, and enhances user productivity.
Group scheduling for meetings and events	Calendar Server improves team collaboration and communication across the organization.
Information sharing through hyperlinks in events or tasks	Calendar Server facilitates collaboration through exchange of information relevant to tasks or events.
Multiple client support	Provides integrated web-based client and support for multiple rich clients including Ximian Evolution and Microsoft Outlook.
Open, modular, and standards-based architecture	Enables customers to deploy customized and personalized solutions.

Making the Communications Services Deployment Highly Available

You can configure Messaging Server and Calendar Server to be highly available by using clustering software. Messaging Server supports both Sun™ Cluster and Veritas Cluster Server software. Calendar Server supports Sun Cluster software. When using clustering software, a secondary Message Server or Calendar Server host provides services to users if the primary system is taken offline for maintenance or is down due to a problem.

Even without the use of Sun Cluster, Messaging Server has built-in monitoring capabilities that continuously check the status of server processes and service availability. Messaging Server can restart process and services automatically, if necessary. Messaging Server logs failures and recovery operations, which you can use for reporting and analysis.

Additionally, you can deploy the Communications Services products in a highly available configuration through use of redundant components. This kind of deployment gives services a high level of uptime. A highly available deployment of this sort requires the redundancy of every component in the service architecture. These components include a duplicate data store server, duplicate network interface cards, and duplicate system storage.

NOTE This guide does not discuss the details of using Sun Cluster in highly available deployments for Communications Services. See the Sun Cluster, Messaging Server, and Calendar Server documentation for more information on this topic.

Using Portal Server with Communications Services

You can install Communications Services products with Portal Server to provide access to messaging and calendar portlets in a portal page. These portlets provide a summary of messaging information, calendar schedules, and address book information. The integration of Portal Server includes single sign-on capabilities between Portal Server, Calendar Express, Messenger Express, and the Communications Express client.

NOTE You can run Communications Express in both Sun Java™ System Schema 1 and Schema 2 environments. If you are using Schema 2, then you can use Access Manager authentication and single sign-on for Communications Express.

Portal Server also supports message archiving for Instant Messaging. In addition, the Messenger Express, Calendar Express, and Instant Messenger clients are made available to users through the Portal Server Desktop.

The following two components of Portal Server provide additional functionality to a basic Communications Services deployment:

- **Portal Server Desktop.** Enables users to access and launch Communications Services applications from portlets.
- **Sun Java™ System Portal Server Secure Remote Access.** Enables remote end users to securely connect to an organization's network and its services over the Internet. End users access Secure Remote Access by logging in to the web-based Portal Server Desktop through the Secure Remote Access gateway. The authentication module configured for Portal Server authenticates the end user. The secure end-user session is established with Portal Server and the access is enabled to the end user's Portal Server Desktop.

NOTE This guide does not discuss portal deploying Communications Services in a portal environment. See the Portal Server documentation for more information.

Understanding the Deployment Process

The Communications Services deployment process consists of the following general phases, referred to as the Solution Life Cycle:

- Analyzing business requirements
- Analyzing technical requirements
- Designing the logical architecture
- Designing the deployment architecture
- Implementing the deployment
- Operating the deployment

The deployment phases are not rigid; the deployment process is iterative in nature. Nevertheless, the following subsections discuss each of the deployment phases independently.

For detailed information on the deployment process for Communications Services, and Java Enterprise System components, see the *Sun Java System Deployment Planning Guide*:

http://docs.sun.com/doc/819-0058

Analyzing Business Requirements

In the business analysis phase, you define the business goal of a deployment project and state the business requirements that must be met to achieve that goal. When stating the business requirements, consider any business constraints that might impact the ability to achieve the business goal. The business analysis phase results in business requirements documents that you later use in the Technical Requirements phase. Throughout the life cycle, you measure the success of your deployment planning, and ultimately your deployed system, according to the analysis performed in the business analysis phase.

Analyzing Technical Requirements

In the technical requirements phase, you start with the business requirements and business constraints defined during the business analysis phase and translate them into technical specifications that can be used to design the deployment architecture. The technical specifications measure quality of service features, such as performance, availability, security, and others.

During the technical requirements phase you prepare the following information:

- Analysis of user tasks and usage patterns
- Use cases that model user interaction with the planned deployment
- Quality of service requirements derived from the business requirements, taking into consideration the analysis of user tasks and usage patterns

The resulting set of usage analysis, use cases, and system requirements documents are inputs to the logical design phase of the Solution Life Cycle. During technical requirements analysis, you might also specify service level requirements, which are the terms under which customer support must be provided to remedy a deployed system failure to meet system requirements. Service level requirements are the basis for service level agreements signed during project approval.

Designing the Logical Architecture

In the logical design phase, you identify the services required to implement the deployment. Once the services are identified, you map logically distinct components providing those services within a logical architecture that shows the dependencies among the components. The logical architecture, together with the technical requirement specifications from the business analysis phase, characterize a *deployment scenario*.

The logical architecture does not specify the actual hardware required to implement the deployment scenario. However, it helps you visualize the interrelationship among components, provides a basis for further analysis of use cases and identified usage patterns, and becomes the starting point for the deployment design phase.

Additional work might be necessary, either in extending services through the use of APIs, or in customizing look and feel, for example, introducing a corporate branding.

For some solutions, development and customization might be quite extensive, requiring you to develop new business and presentation services. In other cases, it might be sufficient to customize existing graphical user interfaces, such as the Portal Server desktop, to achieve the functionality required.

For more information on using product APIs and customizing product functionality, see the appropriate component product documentation, including:

- *Sun Java System Calendar Server Developer's Guide*
  `http://docs.sun.com/doc/819-0025`
- *Sun Java System Communications Services Event Notification Service Guide*
  `http://docs.sun.com/doc/819-0109`
- *Sun Java System Messaging Server Messenger Express Customization Guide*
  `http://docs.sun.com/doc/819-0108`
- *Sun Java System Messaging Server MTA Developer's Reference*
  `http://docs.sun.com/doc/819-0107`

Designing the Deployment Architecture

During the design phase, you map the logical components specified in the logical architecture to physical components in a deployment architecture. You also produce design documents that aid in the implementation of the deployment. Successful deployment design results in the following:

- Project Approval

  Project approval is typically based on design documents created during this phase. During project approval, the cost of the deployment is assessed, and if approved, contracts for implementation of the deployment are signed, and resources to build the project acquired. At what point the actual approval occurs depends on the type of deployment you are designing and internal policies of the company requesting the deployment.

- Deployment Architecture

  The deployment architecture is a high level design document that represents the mapping of logical components to network hardware and software.

- Implementation Specification

  The implementation specification is a set of design documents that includes the following:

  - A detailed design specification used as a blueprint for building out the deployment
  - A user management plan outlining procedures for designing and implementing directory services and data structures needed to provision users for access to system services
  - An installation plan that outlines procedures for distributed installation of the deployment
  - Additional plans covering phased rollout of the deployment, training for end users and administrators, and other plans related to a successful introduction of the deployment

Implementing the Deployment

During implementation phase, you work from design documents created during deployment design to build out the deployment architecture and implement the deployment. Depending on the nature of your deployment project, this phase includes some or all of the following steps:

- Creating and deploying pilot and/or prototype deployments in a test environment
- Designing and running functional tests to measure compliance with system requirements
- Designing and running stress tests to measure performance under peak loads
- Creating a production deployment, which might be phased into production in stages

Once a deployment is in production, you need to continue to monitor, test, and tune the deployment to ensure that it fulfills the business goals.

Chapter 2

Analyzing Your Communications Services Requirements

Planning your Communications Services 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 Communications Services design.

This chapter contains the following sections:

- Identifying Deployment Goals
- Determining Project Goals

For detailed information on the deployment process for Communications Services, and Java Enterprise System components, see the *Sun Java System Deployment Planning Guide*:

`http://docs.sun.com/doc/819-0058`

Identifying Deployment Goals

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

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

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

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

Defining Business Requirements

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

Operational Requirements

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

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

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

Culture and Politics

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

- Some sites might require their own management of the deployed solution. Such demands can raise the project's training costs, complexities, and so forth.

- Given that the LDAP directory contains personnel data, the Human Resources department might want to own and control the directory.

Defining Technical Requirements

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

Supporting Existing Usage Patterns

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

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

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

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

Site Distribution

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

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

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

Network

Here are some questions to 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 if you move them?
- Do you want to reduce network HTTP traffic?

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

Existing Infrastructure

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

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

Support Personnel

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

- Is there sufficient capacity in operations and technical support groups to facilitate this deployment?

- Can operations and technical support groups cope with the increased load during deployment phase?

Defining Financial Requirements

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

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

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

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

Defining Service Level Agreements (SLAs)

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

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

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

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

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

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

Planning for Growth

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

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

Ideally, you should design your architecture to easily accommodate future growth. For example, use logical names for the Communications Services themselves. See "Using Logical Service Names" on page 95 for more information. 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 your Communications Services. 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.

Chapter 3

Understanding Product Requirements and Considerations

This chapter describes requirements and considerations that impact the design of your deployment. You need to understand these requirements and considerations to accurately determine your Communications Services architecture.

This chapter contains the following sections:

Planning for Various Components

When designing your Communications Services deployment architecture, take into account the requirements of the various component products of your deployment. For example, if you have a technical requirement to integrate Communications Services with other Java System products, you need to choose your schema accordingly. Inter-product dependencies, for example, how Communications Services access and place load on Directory Server, present deployment choices as well.

Understanding the individual components of each product enables you to plan for the type of architecture to best suit your requirements. Depending on your deployment, you need to potentially understand and plan for the following components:

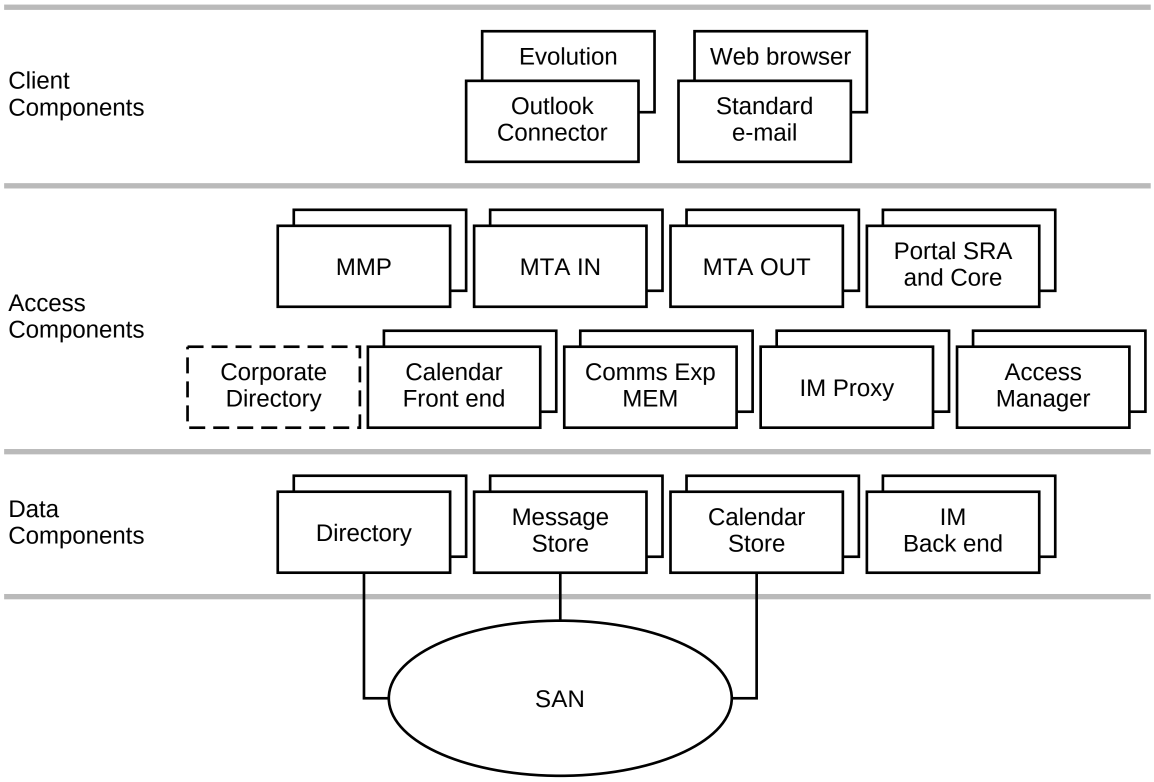
- LDAP Directory Information Tree
- Schema
- Directory Server (Access Manager)
- Messaging Server
    - Message Transfer Agent (MTA)
    - Message Store
    - MMP (Messaging Multiplexor)
    - MEM (Messaging Express Multiplexor)
- Calendar Server
    - Front End
    - Calendar Store
- Instant Messaging
    - Instant Messaging Proxy
    - Instant Messaging Back End
- Portal Server
- Connector for Microsoft Outlook
- Communications Express

Understanding Service Components and Service Tiers

When planning a Communications Services deployment for multiple component products or services, you need to understand the composition of each component product (or service) itself.

Figure 3-1 on page 56 illustrates how you can separate each service into components that can be deployed on separate hosts, and the particular tier each component occupies. Though you can deploy all components on a single host, or deploy a particular service's components on the same host, consider moving to a tiered architecture. A tiered architecture, whether it be single-tiered, or two-tiered, provides a number of benefits. See "Benefits of a Single-tiered Architecture" on page 90 and "Benefits of a Two-tiered Architecture" on page 91 for more information.

Figure 3-1 Communications Services Components



In the preceding figure, the client components consist of the Outlook Connector plugin, thick clients such as Evolution, browsers, and standard email applications. These components reside on end users' client computers. The access layer components consist of front-end services from Messaging Server (MMP, MTA, and MEM); Calendar Server; Communications Express (which must be collocated with MEM); Instant Messaging (Instant Messaging Proxy); Portal Server (SRA and Core); Access Manager for authentication, and a corporate directory, which provides address book lookup. The data layer components consist of back-end services from Directory Server (which, in itself, can consist of front-end and back-end components); Messaging Server (Message Store); Calendar Server (Calendar Store); and Instant Messaging. A Storage Area Network (SAN) "cloud" represents the physical data storage.

NOTE The corporate directory shown in this figure is not a component product in itself. It represents a “copy” of the corporate directory that enterprises typically deploy in the access layer for clients to perform address-book type lookups.

The following sections explain these various components in more detail.

LDAP Directory Information Tree Requirements

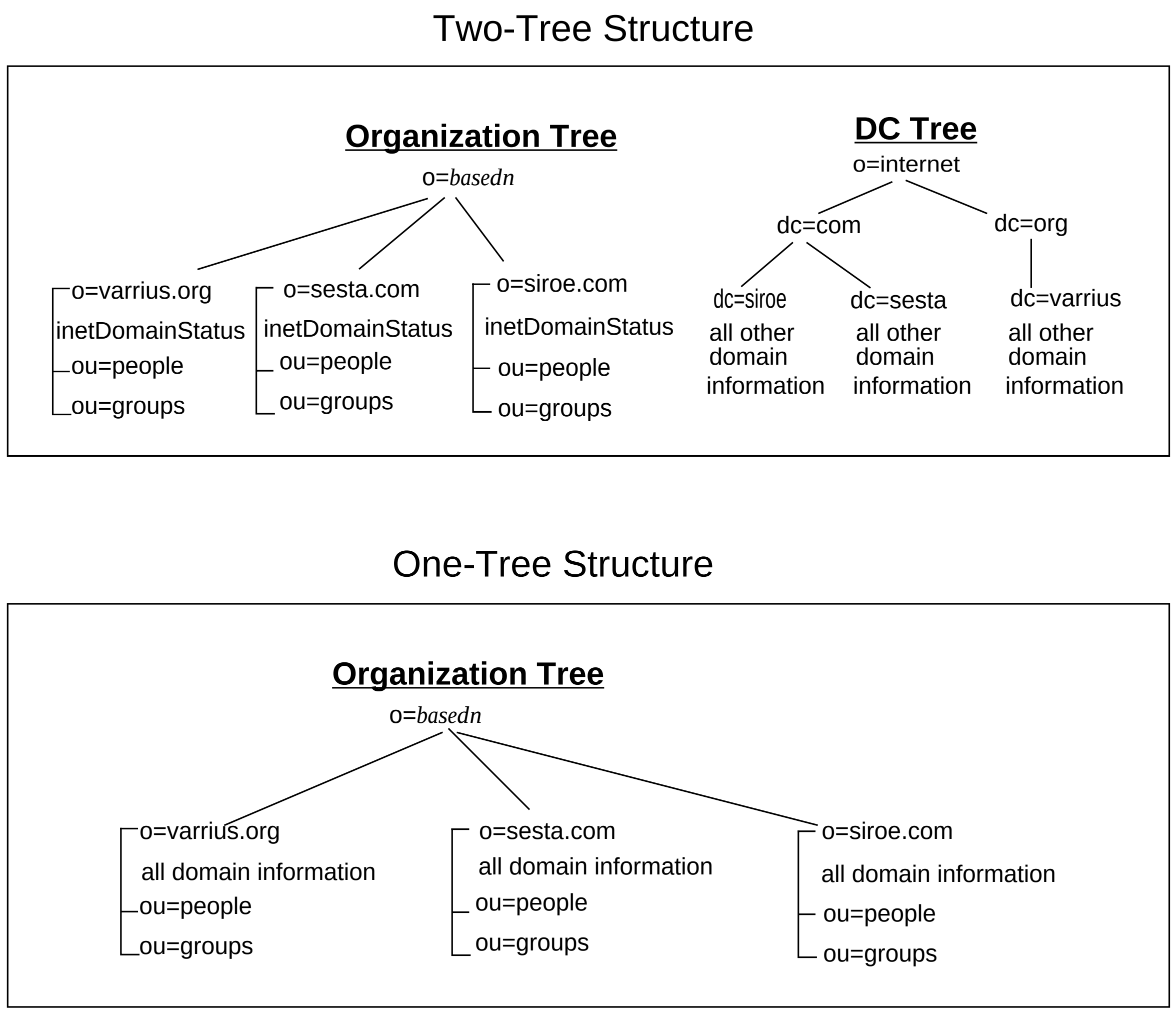
The Directory Information Tree (DIT) is a way to organize directory entries in a tree structure, or schema, with nodes representing domains, subdomains, users, and groups. Sun Java Enterprise System introduces a fundamental change to how the directory is structured by implementing a one-tree structure.

Changes in the DIT Structure

Messaging Server and Calendar Server have introduced a one-tree structure, where there is no Domain Component (DC) Tree. All domain information is held in domain nodes in the Organization Tree. Aliasing is handled entirely differently in the new one-DIT structure.

The bottom half of Figure 3-2 illustrates a one-tree LDAP structure.

Figure 3-2 Two-Tree LDAP Structure Compared With One-Tree Structure



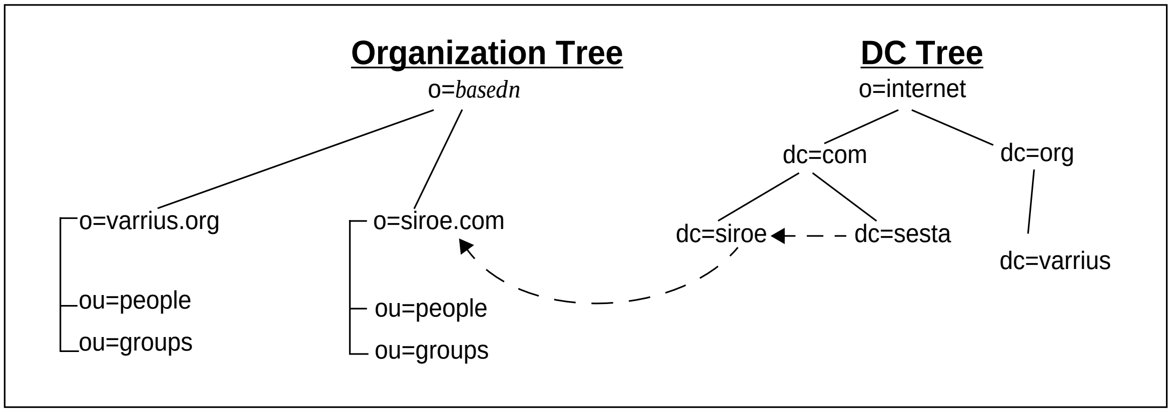
Benefits of a One-Tree DIT Structure

The main advantages to using the one-tree structure Schema 2 native mode are:

- The structure is integrated with Access Manager.
- The structure is more closely aligned with industry standards.
- The structure is significantly less complex than the two-tree structure.

As illustrated in the following figure, in the two-tree structure, some nodes point directly to a node in the Organization Tree (using the attribute `inetDomainBaseDN`). Other nodes are aliased nodes, which instead of pointing directly to an Organization Tree node, point to another DC Tree node, using the `aliasedObjectName` attribute.

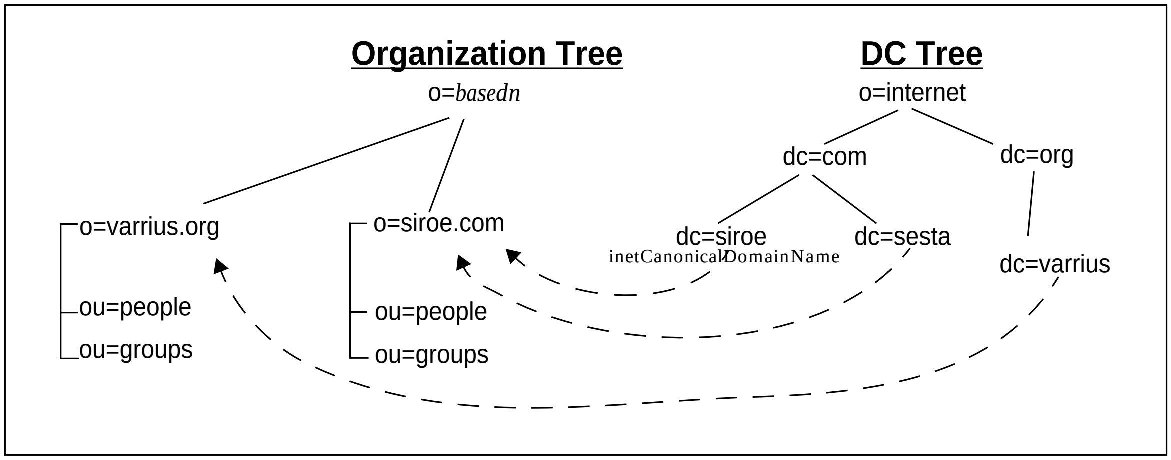
Figure 3-3 Two-Tree Aliasing With `aliasedDomainName` and `inetDomainBaseDN`



In the previous figure, `sesta.com` in the DC Tree points to `siroe.com` in the DC Tree using `aliasedObjectName`, and `siroe.com` points to the like named node in the Organization Tree, using `inetDomainBaseDN`.

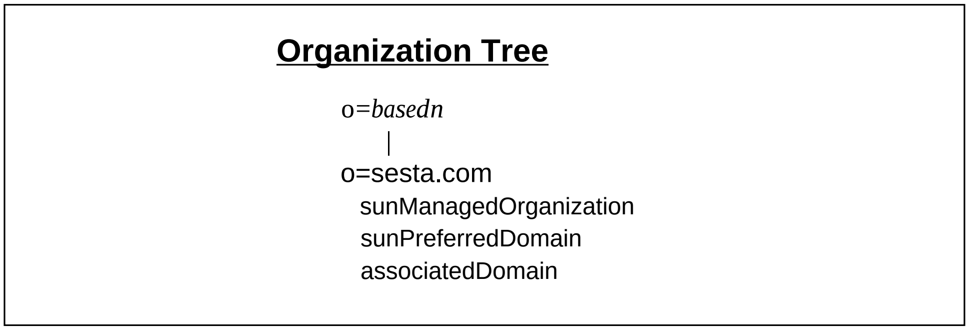
Furthermore, as shown in Figure 3-4 on page 60, there could be one or more nodes in the DC Tree using `inetDomainBaseDN` to point directly to the same node in the Organization Tree. In this case, a "tie-breaker" attribute, `inetCanonicalDomainName`, is necessary on one of the DC Tree nodes to designate which is the "real" domain name (the domain where the mail actually resides and where the mail is routed).

Figure 3-4 Two-Tree Aliasing With `inetCanonicalDomainName`



By contrast, a one-tree structure contains only an Organization Tree, as shown in the following figure.

Figure 3-5 One-Tree Aliasing With `associatedDomain`



In the one-tree structure, domain nodes in the Organization Tree contain all the domain attributes formerly found on the DC Tree. Each domain node is identified by the `sunManagedOrganization` object class and `sunPreferredDomain` attribute, which contains the DNS domain name. A domain node can also have one or more `associatedDomain` attributes, which list the alias names this domain is known by. Contrary to the two-tree structure, there are no duplicate nodes for the alias names.

A one-tree DIT structure is beneficial in how you partition data for organization-specific access control. That is, each organization can have a separate subtree in the DIT where user and group entries are located. Access to that data can be limited to users in that part of the subtree. This allows localized applications to operate securely.

In addition, for new deployments of Calendar Server or Messaging Server, a one-tree structure maps better to existing single-DIT LDAP applications.

Schema Requirements

Before you install any of the Communications Services products, you need to understand which schema you will use. The schema is the set of definitions describing what types of information can be stored as entries in the directory. Two schema choices, Sun Java™ System LDAP Schema 1 and Sun Java™ System LDAP Schema 2, are available and supported with Communications Services. Your choice of schema depends on the following criteria:

Use Schema 2 if you:

- Want to use Access Manager 6 2005Q1 (formerly Identity Server), either for user provisioning or single sign-on (SSO)
- Are installing Communications Services components for the first time
- Are integrating Communications Services with other Java Enterprise System products, such as Portal Server

NOTE You do not have to use Access Manager 6 to provide SSO. If you choose, you can still use the trusted circle type of SSO, which does not rely on Access Manager 6.

Use Schema 1 if you:

- Have an existing version of any of the Communications Services components, for example, if you are upgrading from Messaging Server 5.2
- Do not need to provision users through Access Manager (unless Access Manager SSO is also a requirement)
- Want to use Sun ONE Delegated Administrator (formerly called iPlanet™ Delegated Administrator) to do your user provisioning

See Chapter 8, "Understanding Schema and Provisioning Options" for more information on schema choices.

Directory Server Considerations

Sun Java System Directory Server provides flexible, multi-tiered data storage for intranet, network, and extranet information. Directory Server integrates with existing systems and acts as a centralized repository for the consolidation of employee, customer, supplier, and partner information. You can extend Directory Server to manage user profiles and preferences, as well as extranet user authentication.

All custom LDAP schemas, such as those from Portal Server, Access Manager, Messaging Server, Calendar Server, and Instant Messaging Server, install into a single Directory.

There are many ways to architect your data environment and many factors to consider that depend on your business objective and expected usage patterns. Your directory design should address the following areas:

- Directory schema and object class definitions
- The Directory Information Tree (DIT)
- Groups and membership definitions
- A strategy for Access Control Lists (ACLs) including static and dynamic groups
- Directory architecture for high availability and performance, including failover, replication, and referrals
- Management of the directory

See the *Sun Java System Directory Server Deployment Planning Guide* for a complete description of these factors and suggestions about how to architect your data environment:

`http://docs.sun.com/doc/817-7607`

Directory Server and Tiered Architecture Considerations

In moving from a single-tiered architecture to a multiple-tiered architecture, Directory Server should be the first component that you "split out" onto its own machine. At a certain point of load, Directory Server and Messaging Server on the same host have inherent performance impacts. This is due to the way Messaging Server is architected to work with Directory Server. Separating Directory Server onto its own machine is the first step to improve performance for a deployment.

See Chapter 5, "Developing a Communications Services Logical Architecture" for more information on tiered architectures.

NOTE You can install Directory Server in such a way to have a clear separation between the directory user management and the software application configuration. In this architecture, there are two directories: a user/group directory on a directory host, and a configuration directory on a separate host. Should you want to remove the software application configuration piece, this separation provides a cleaner way of removing that information from Directory Server.

Directory Server Topology Considerations

Though it is feasible to build a deployment around an instance of Directory Server installed on a single machine, the other Communications Services components depend upon the directory as a core service to function. Thus, beyond trivial deployments, you should plan to deploy Directory Server in a redundant or highly availability configuration.

The first step toward making Directory Server more available is to establish a pair of master directory servers. Next, multimaster replication can be used to improve the LDAP write throughput and availability. If Sun Cluster is used for a high-availability deployment, then the two LDAP masters are clustered together. See "Directory Server and High Availability" on page 99 for more information.

Directory Server Capacity Planning

While there are no hard and fast rules for Directory Server capacity planning, actively monitoring the directory is essential to ensure that performance metrics are met. When the system is not meeting these metrics, then it is time to add an additional directory consumer. Typically, you will want to monitor:

- Hit load
- Cache load
- Requests per second

Evaluate the above metrics against a target response time of 10 Milliseconds. The IOWAIT should not exceed 10 Milliseconds, and the sum of the CPU utilization in this tier should not exceed 70 percent.

Directory Server and Calendar Server Interaction Considerations

Calendar Server performs multiple writes to user entries stored in Directory Server. The bulk of these writes occur when the user logs into Calendar Server for the first time and when the user performs certain actions. These actions include creating a calendar, subscribing to a calendar, changing a preference, and so on. If you do not take these actions into consideration, the Directory Master Server can experience heavy loads.

If you use Directory replication, the LDAP Master Server is replicating entries to the LDAP Replica servers. As Calendar users perform one of these actions, Calendar Server will only be able to write changes to the Master Directory Server. This is because the Replicas are read-only.

A second interaction consideration exists in these replicated Directory structures. As users make preference changes, their changes might not be rendered successful until the change is successfully replicated from the Master Directory Server to the Directory Replica, which is in use by the Calendar Server. A workaround is available, in which you configure Calendar Express (`cshttpd`) attempts to cache the change locally to avoid this latency delay. See "Planning for the Calendar Server LDAP Data Cache" on page 268 for more information.

Directory Server and Personal Address Book Considerations

The Messenger Express Client supports the concept of a Personal Address Book (PAB). This enables users to store personal contacts (for example, business contacts, friends, and family) in the Directory Server. Each time a new personal contact is added to the user's PAB, a write is made on the Directory Server. If you do not take these actions into consideration, the LDAP Master Server can face heavy loads (regardless of the Directory replication strategy).

One method to avoid performance issues on the User and Group Directory Server is to place the PAB information on a separate Directory Server. This enables PAB interactions to avoid placing a load on the LDAP Master Server.

NOTE If you are running both the current Communications Express client and also the deprecated Messenger Express Web mail interface, the address books used by these two clients do not share information. If end users switch between the two client interfaces, the two address books will contain different entries.

Messaging Server Considerations

In developing your Communications Services architecture, you need to evaluate the performance aspects of the following Messaging Server components:

- Message Store
- Message Transfer Agent (MTA)
- Mail Message Proxy (MMP)
- Messaging Express Multiplexor (MEM)

For a complete discussion of the performance aspects of these components, and potential hardware solutions, see "Performance Considerations for a Messaging Server Architecture" on page 179.

Calendar Server Considerations

Calendar Server consists of five major services:

- HTTP Service (`cshttpd`) listens for HTTP requests. It receives user requests and returns data to the caller.
- Administration Service (`csadmind`) is required for each instance of Calendar Server. It provides a single point of authentication and administration for the Calendar Servers and provides most of the administration tools.
- Notification Service (`csnotify`) sends notifications of events and to-dos using either email or the Event Notification Service.
- Event Notification Service (`enpd`) acts as the broker for event and alarm notifications.
- Distributed Database Service (`csdwpd`) links multiple database servers together within the same Calendar Server system to form a distributed calendar store.
- Backup Service (`csstored`) implements automatic backups, both archival backups and hot backups. The first backup is a snapshot with log files, the second is a snapshot with log files applied. This service is automatically started when you run the `start-cal` command. However, it is not enabled at installation time, so you must configure it to function. If left unconfigured, Backup Service sends out a message to the administrator every 24 hours, with the notification that the service is not configured.

In a scalable Calendar Server deployment, you would deploy front-end systems in conjunction with a back-end server. The front-end systems would contain one instance of the `cshttpd` daemon per processor and a single Administration Service. A back-end server would contain an instance of Notification Service, Event Notification Service, Distributed Database Service and Administration Service.

Authentication and XML / XSLT transformation are two Calendar Service activities that generate heavy load. Additional CPUs can be added to meet quality of service requirements. In a scalable environment, these heavy load activities take place on the front-end system(s), permitting more CPUs to be added to individual front-end systems, or more front-end systems to be added, to meet quality of service requirements.

NOTE The preceding paragraph is not applicable if the Communications Express Calendar client is used for calendar access. Communications Express uses the WCAP protocol to access Calendar Server data and therefore the Calendar Server infrastructure is not doing the XML / XSLT translations. See Part V, "Deploying Communications Express" for information on deploying Communications Express.

Calendar back-end services usually require half the number of CPUs sized for the Calendar front-end services. To support quality of service by the Calendar front-end system, the Calendar back-end system should use around two-thirds of the front-end CPUs.

Consider early on in your deployment planning to separate the Calendar Service into front-end and back-end services.

The Calendar Server HTTP process that is typically a component of the front-end services is a dominant user of CPU time. Thus, account for peak calendar usage and choose sufficient front-end processing power to accommodate the expected peak HTTP sessions. Typically, you would make the Calendar Server front end more available through redundancy, that is, by deploying multiple front-end hosts. As the front-end systems do not maintain any persistent calendar data, they are not good candidates for HA solutions like Sun Cluster or Veritas. Moreover, the additional hardware and administrative overhead of such solutions make deploying HA for Calendar Server front ends both expensive and time-consuming.

NOTE The only configuration for Calendar front ends that might warrant a true HA solution is where you have deployed the Calendar front end on the same host that contains a Messaging Server MTA router. Even in this configuration, however, the overhead of such a solution should be carefully weighed against the slight benefit.

A good choice of hardware for the Calendar Server front ends is a single or dual processor server. You would deploy one instance of the Calendar Server `cshttpd` process per processor. Such a deployment affords a cost-effective solution, enabling you to start with some level of initial client concurrency capability and add client session capacity as you discover peak usage levels on your existing configuration.

When you deploy multiple front ends, a load balancer (with sticky / persistent connections) is necessary to distribute the load across the front-end services.

NOTE Communications Express does not scale beyond two processors. The same hardware choices explained previously for Calendar Server apply to Communications Express deployments.

The Calendar Server back-end services are well balanced in resource consumption and show no evidence of bottleneck formation either in CPU or I/O (disk or network). Thus, a good choice of hardware for the back end would be a SPARC server with a single striped volume. Such a machine presents considerable capacity for large-peak calendar loads.

If your requirements include high availability, it makes sense to deploy the Calendar Server back end with Sun Cluster, as the back end does contain persistent data.

NOTE In a configuration with both front-end and back-end Calender Server hosts, all hosts must be running:

- The same operating system
- The same releases of Calendar Server, including patch or hotfix releases

Instant Messaging Considerations

As with other Communications Services components, you can create an architecture in which you separate Instant Messaging into front-end (Instant Messaging multiplexor) and back-end components (server and store). See "Developing Instant Messaging Architectural Strategies" on page 305 for more information.

Portal Server Considerations

You can install Communications Services products with Portal Server to provide an “umbrella” front end to access messaging, calendar, and instant messaging applications. The integration of Portal Server includes single sign-on capabilities between Portal Server, Calendar Express web client, Messaging Express web client and Communications Express client. In addition, the Messaging Express, Calendar Express, and Instant Messaging clients are made available to users through the Portal Server desktop.

See the *Sun Java System Portal Server Deployment Guide* and the *Sun Java System Access Manager Deployment Planning Guide* for more information:

`http://docs.sun.com/doc/817-6257`

`http://docs.sun.com/doc/817-7644`

Connector for Microsoft Outlook Considerations

This section describes some deployment issues you will encounter deploying Connector for Microsoft Outlook. For complete information, see the Connector for Microsoft Outlook documentation:

`http://docs.sun.com/db/coll/MessagingServer_05q1`

Sun Java System Connector for Microsoft Outlook enables Outlook to be used as a desktop client with Sun Java Enterprise System. Connector for Microsoft Outlook is an Outlook plug-in that must be installed on the end-user’s desktop. Connector for Microsoft Outlook queries Messaging Server for folder hierarchies and email messages. It converts the information into Messaging API (MAPI) properties that Outlook can display. Similarly, it uses the WCAP protocol to query Calendar Server for events and tasks which are then converted into MAPI properties. With this model, Connector for Microsoft Outlook builds an end-user Outlook view from two separate information sources: mail from Messaging Server and calendar information from Calendar Server.

When users create and modify items through Outlook, Connector for Microsoft Outlook passes the new message along to the appropriate server depending on its message type. It sends new outgoing email to an SMTP mail server for delivery, and sends modified email messages back to the user’s IMAP folder for storage. New calendar events and tasks are converted into a standard format to be stored in the Calendar Server database.

Connector for Microsoft Outlook Component Product Dependencies

To use Connector for Microsoft Outlook, both Messaging Server and Calendar Server are required in the same deployment. See those products' release notes for information on supported versions.

For Connector for Microsoft Outlook to function correctly, the following LDAP attributes in the Sun Java System Directory Server should be indexed for at least presence and equality to improve the overall performance:

- `icsCalendar`
- `mail`
- `mailalternateaddress`

See the *Sun Java System Connector for Microsoft Outlook Release Notes* for a complete list of product dependencies.

`http://docs.sun.com/db/coll/MessagingServer_05q1`

Migrating Sun ONE Calendar Server Data

If you are using a version of Calendar Server prior to Sun ONE Calendar Server 6.0, you need to engage Sun Client Services to convert and migrate the data to the new format. This Calendar Server data migration is required for the use of Outlook, and is necessary because of the underlying changes in the storage and management of recurring events. No migration service is required for new customers of Sun Java System Calendar Server.

Migrating Exchange Server Data

Connector for Microsoft Outlook does not convert Microsoft Exchange Server messages on the Exchange Server. You need to engage with Sun Client Services to convert the data.

Communications Express Considerations

Sun Java System Communications Express provides an integrated web-based communications and collaboration client. Communications Express is a common part of Messaging Server and Calendar Server, providing end users with a web interface to their calendar information and mail, as well as an address book.

Communications Express consists of three client modules: Calendar, Address Book, and Mail. The Calendar and Address Book client modules are deployed as a single application on a web container. Messenger Express is the standalone web-based mail application that uses the HTTP service of the Messaging Server. Messenger Express should be deployed on the same system as Communications Express.

Communications Express depends upon the following Sun Java System component products:

- Directory Server
- Access Manager (If you are using Sun Java System LDAP Schema Version 2)
- Calendar Server
- Messaging Server
- Web Server / Application Server (for web container)

S/MIME Considerations

Communications Express Mail includes the security advantages of the Secure / Multipurpose Internet Mail Extension (S / MIME). Communications Express Mail users who are set up to use S / MIME can exchange signed or encrypted messages with other Communications Express Mail users, and with users of the Microsoft Outlook mail system.

See "Requirements for Using S / MIME with Communications Express Mail" on page 350 for more information.

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 Communications Services 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
- Understanding Network Infrastructure Components
- Planning Your Network Infrastructure Layout

Understanding Your Existing Network

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

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 software installation to the minimum number of packages necessary to support the applications to be hosted. Achieving minimization in services, libraries, and applications helps increase security by reducing the number of subsystems that must be maintained. The Solaris™ Security Toolkit provides a flexible and extensible mechanism to minimize, harden, and secure Solaris systems.

Your Site Security Policy should provide direction on such issues.

Load Balancers

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

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

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

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

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

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?

Domain Name System (DNS)

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

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

Planning Your Network Infrastructure Layout

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

- DMZ
- Intranet
- Internal network
- Proxies
- Firewall Configuration
- Mobile users

Demilitarized Zone (DMZ)

These days, most company networks are configured for a DMZ. The DMZ separates the corporate network from the Internet. The DMZ is a tightly secured area into which you place servers providing Internet services and facilities (for example, web servers). These machines are hardened to withstand the attacks they might face. 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. In addition, just as multiple DMZs can be beneficial—depending on your services and your network's size—multiple intranets might also be helpful.

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

Internal Network

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

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

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

Developing a Communications Services Logical Architecture

This chapter describes how to develop your Communications Services logical architecture. The logical architecture is a design that depicts the logical building blocks of the Communications Services components and the infrastructure services needed to support them.

This chapter contains the following sections:

- Communications Services Deployment Logical Architectures Overview
- Horizontal Scalability Strategy
- Other Deployment Issues

Communications Services Deployment Logical Architectures Overview

You can deploy Communications Services in either a *single-tiered* or *two-tiered* logical architecture. Deciding on your logical architecture is crucial, as it determines which machine types you will need, as well as how many.

In general, enterprise corporate deployments use a single-tiered architecture while internet service providers (ISPs) and telecommunications deployments use a two-tiered architecture. However, as with all generalities, the exceptions prove the rule. Small ISPs might just as well deploy on a single machine, and larger, centralized enterprises might deploy in a two-tiered architecture for many of the same reasons that ISPs do. As more and more corporations look to offer ease of access to employees working remotely, their deployments will begin to look more and more like an ISP.

This section discuss the following Communications Services logical architectures:

- **Single-tiered Architecture.** All services are either located on a single host sufficiently provisioned with memory and CPUs, or deployed on multiple servers, with each server hosting all of a particular component product's services.
- **Two-tiered Architecture.** The access and data layers for the component products are separated onto different servers.
- **Edge Architecture.** Builds on the two-tiered architecture by also providing for secure connections over the Internet to a mobile workforce.

Whether you deploy Communications Services in a single-tiered or multiple-tiered architecture, you need to understand the advantages and disadvantages of both models.

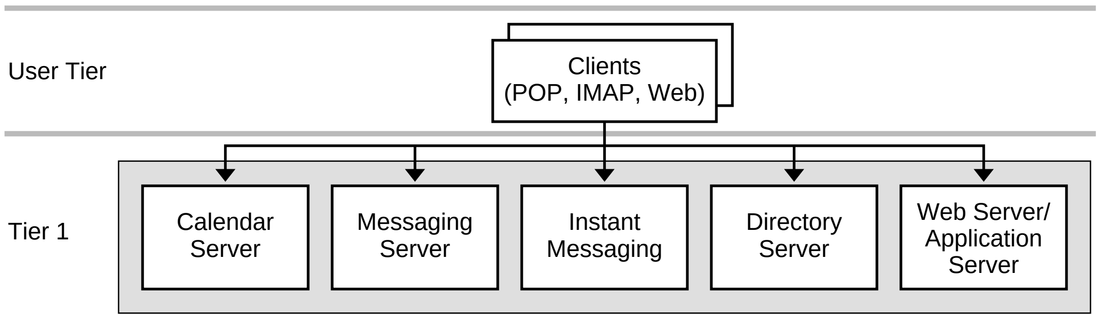
Single-tiered Logical Architecture for One Host

As its name implies, the single-tiered logical architecture for one host locates all services onto a single machine. In general, such an architecture is best suited for enterprises that are:

- Composed of 500 users or less
- Not geographically distributed
- Served by few administrators
- Seeking an entry-level configuration

The following figure represents the single-tiered logical architecture for one host.

Figure 5-1 Single-tiered Architecture for One Host



End-user client programs, such as Outlook and Messenger Express, form the User Tier. Tier 1 is a single machine running all services, including messaging, calendar, instant messaging, and directory. If you deploy Communications Express, the single machine is also running Web Server (or Application Server). The distinction of the single tier deployment is that end users communicate directly to the stores, and not through proxies or other agents.

The single-tiered logical architecture for one host requires a machine that provides sufficient CPU, memory, and storage. You should work with your Sun representative to determine the machine that best meets your organization’s needs for this type of deployment.

When implementing the single-tiered logical architecture for one host, you can position the deployment for growth into a multi-tiered architecture by assigning logical names to services. Such a configuration makes use of DNS mapping to direct users to the same front-end process (machine). If, in the future, you need to make changes to accommodate growth, such as splitting out services in a tiered fashion, users do not need to reconfigure their client applications. See “Using Logical Service Names” on page 95 for more information.

Single-tiered Logical Architecture for Multiple Hosts

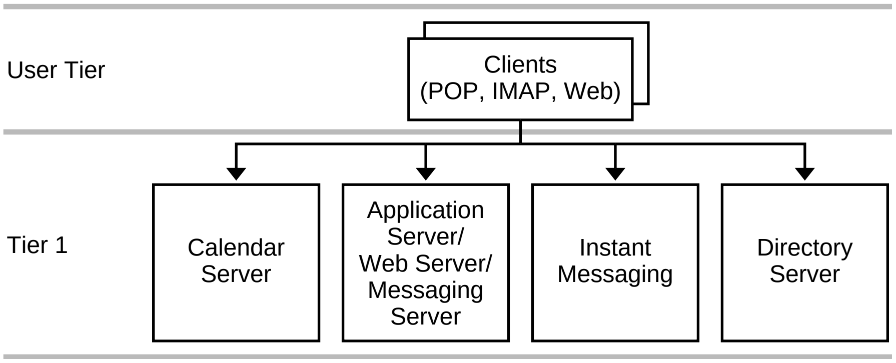
The single-tiered logical architecture for multiple hosts is a set of servers that each run the services particular to a component product. For example, the Messaging Server host is installed and configured to run all the Messaging Server services, the Calendar Server host is installed and configured to run all the Calendar Server services, and so on. This architecture might also be configured for high availability.

The distinction of the single-tiered logical architecture is that end users communicate directly to the data stores, and not through proxies or other agents. For example, in Messaging Server, users would not be routed through MMPs or MTAs. The single-tiered logical architecture might have standalone MTA routers for routing mail between servers, or in and out of the corporate network, but end users submit mail to the MTA on their message stores. No MMPs are involved in intranet connection to the message stores.

The same idea applies to both Calendar Server and Instant Messaging. In the single-tiered logical architecture, no front-end processes are located on separate machines.

Figure 5-2 on page 84 represents the single-tiered logical architecture for multiple hosts.

Figure 5-2 Single-tiered Architecture for Multiple Hosts



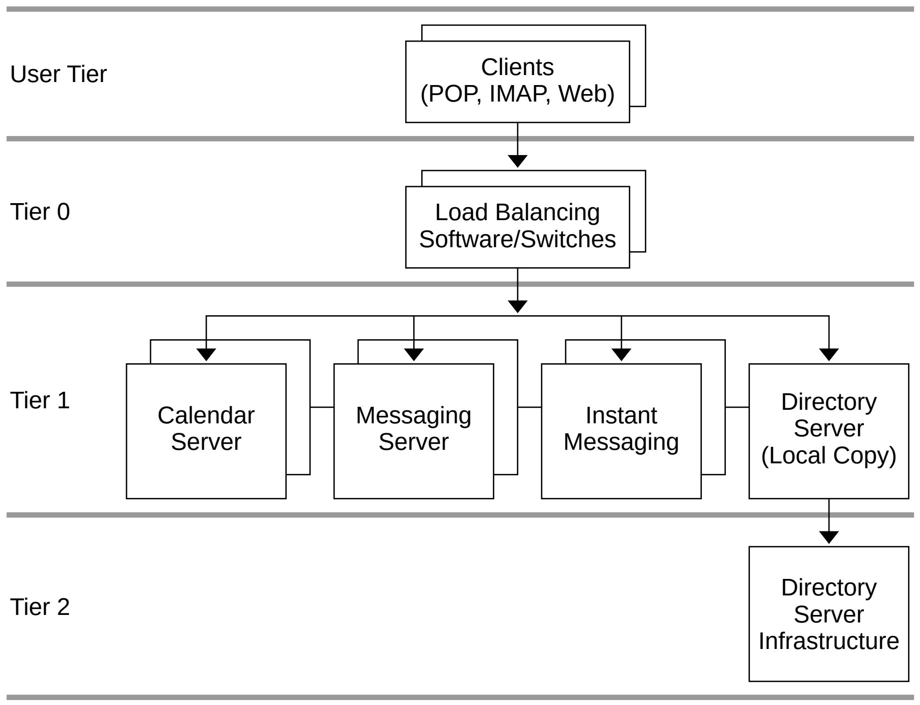
In the preceding figure, end-user client programs, such as Outlook and Messenger Express, form the User Tier. Tier 1 is a set of four servers. One server runs the Calendar Server processes, the second runs the Messaging Server processes, the third runs the Instant Messaging processes, and the fourth runs the Directory Server process. If you are deploying Communications Express, the Messaging Server host also includes a web server, either Web Server or Application Server, (for Webmail).

Single-tiered Distributed Logical Architecture

The single-tiered distributed logical architecture is a variant of the single-tiered architecture in that the Directory Server is deployed in two tiers. Such a deployment works well for enterprises with small departments or organizations that are geographically distributed. Each department or office has its own services (mail, calendar, instant messaging) and a local directory instance (consumer). All the local directory instances are cached, but are synchronized with the centralized, master corporate repository. This is a fairly common scenario for offices with low bandwidth connectivity. The directory is architected in a two-tiered fashion and replicated over the low-bandwidth to keep data local.

Figure 5-3 on page 85 represents the single-tiered distributed logical architecture.

Figure 5-3 Single-tiered Distributed Architecture



In the preceding figure, end-user client programs, such as Outlook and Messenger Express, form the User Tier. Tier 0 consists of load balancers that distribute load across the Tier 1 layer. Tier 1 is a set of multiple servers for the Communications Services processes. Multiple servers run the Calendar Server processes, multiple servers run the Messaging Server processes, and multiple servers run the Instant Messaging processes. Directory Server processes are split between a consumer server in Tier 1 running a local, replicated copy of the directory, and another server situated in Tier 2, which contains the master copy of the directory. Notice that in this kind of deployment, client queries are directed to the local directory copy, not to the master copy. Only the local Directory Server communicates to the master Directory Server.

NOTE When deploying a single-tiered architecture with Internet connectivity, use a separate access layer. For example, you direct access to the data stores from inside the intranet without having to use SSL. However, you direct access to the data stores from the Internet through an access layer over SSL. This offloads much of the SSL load on the data stores to the access layer that separates it from the Internet.

The downside to this type of deployment is that users who make use of the server from a system that is sometimes on the corporate intranet and sometimes accessing the server from the Internet must configure their client applications to use SSL all the time. This is because it is too much trouble to switch back and forth. Therefore, there will still be a substantial percentage of SSL traffic being put on the stores directly. By using an access layer inside the intranet, you can remove that problem and by limiting connection directions further protect the intranet from illegal access.

Two-tiered Logical Architecture

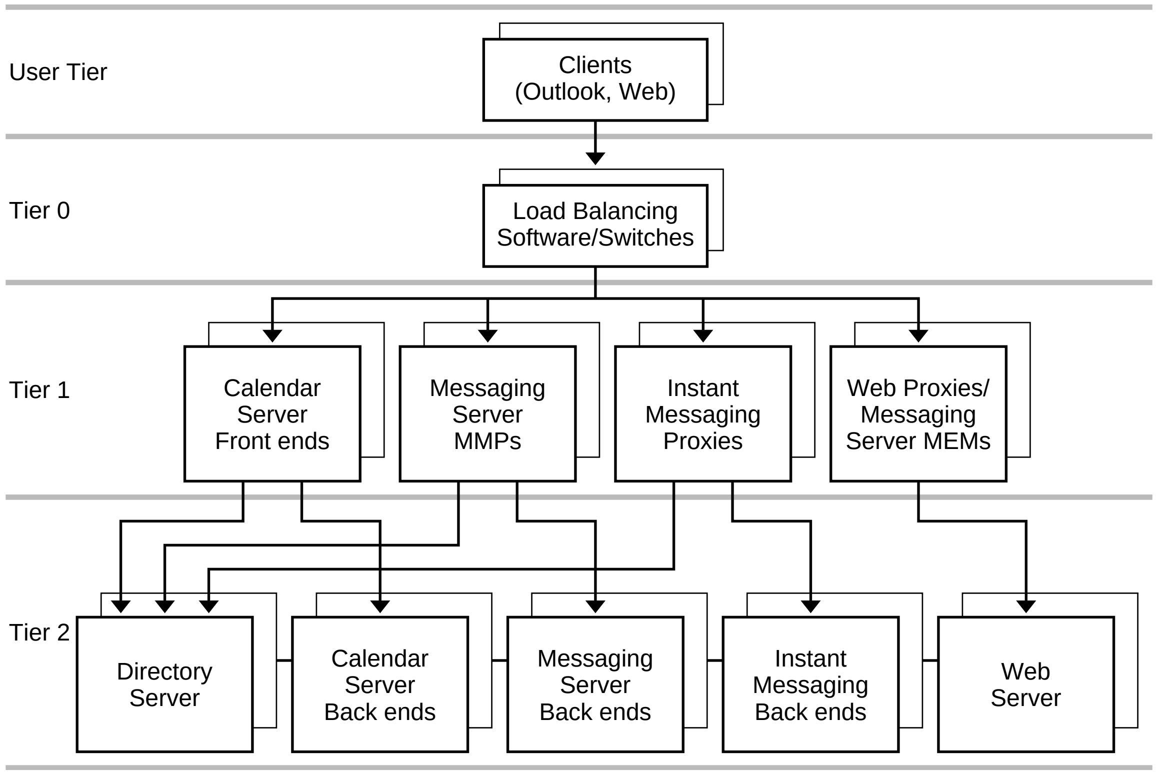
In a two-tiered logical architecture, the data stores communicate through front-end processes. In the case of Messaging Server, this means MMPs, MEMs, and MTAs are residing on separate machines from the data store processes. A two-tiered architecture enables the mail store to offload important and common tasks and focus on receiving and delivering mail. In the case of Calendar Server, this means the HTTP service and Administration service reside on a separate machine from the store processes. In the case of Instant Messaging, this means the proxy service is residing on a separate machine from the back-end processes.

There might be some level of cohabitation with other services. For example, you could have the Calendar store and the Message Store on the same machine. Similarly, you could have the calendar front end on the MMP machine.

In a two-tiered logical architecture, Directory Server is usually a complex deployment in its own right, with multi-master and replication to a set of load-balanced consumer directories.

Figure 5-4 on page 87 represents the two-tiered logical architecture.

Figure 5-4 Two-tiered Architecture


User Tier
Clients (Outlook, Web)
Tier 0
Load Balancing Software/Switches
Tier 1
Calendar Server Front ends
Messaging Server MMPs
Instant Messaging Proxies
Web Proxies/ Messaging Server MEMs
Tier 2
Directory Server
Calendar Server Back ends
Messaging Server Back ends
Instant Messaging Back ends
Web Server

In the preceding figure, end-user client programs, such as Outlook and Messenger Express, form the User Tier. The load balancers form Tier 0. The Calendar Server, Messaging Server, Instant Messaging, and web proxy and MEM front ends form Tier 1. Finally, the Directory Server, Calendar Server, Messaging Server, and Instant Messaging back ends form Tier 2. When deploying Communications Express, you could have Web Server in Tier 2 as well.

A two-tiered architecture enables you to deploy Tier 1 and Tier 2 elements as separate instances, increasing overall flexibility of design. Additionally, you enhance system security by assigning discrete functions to individual instances.

For typical deployments, place the messaging and calendar front ends within the network Demilitarized Zone (DMZ), connecting to the main messaging and calendar services through a firewall. This configuration enables you to scale the system horizontally, as the Tier 1 elements can be scaled independently. Do not scale these elements beyond the capacity of the back-end servers.

When the front-end elements have reached the capacity of the back-end servers, you can scale the back-end Tier 2 elements to support more users. In general, the front end should scale as a function of the traffic. The back end should be scaled as a function of the number of users.

NOTE For specific instructions on sizing components in single-tiered or two-tiered architectures, contact your Client Services representative.

Edge Logical Architecture

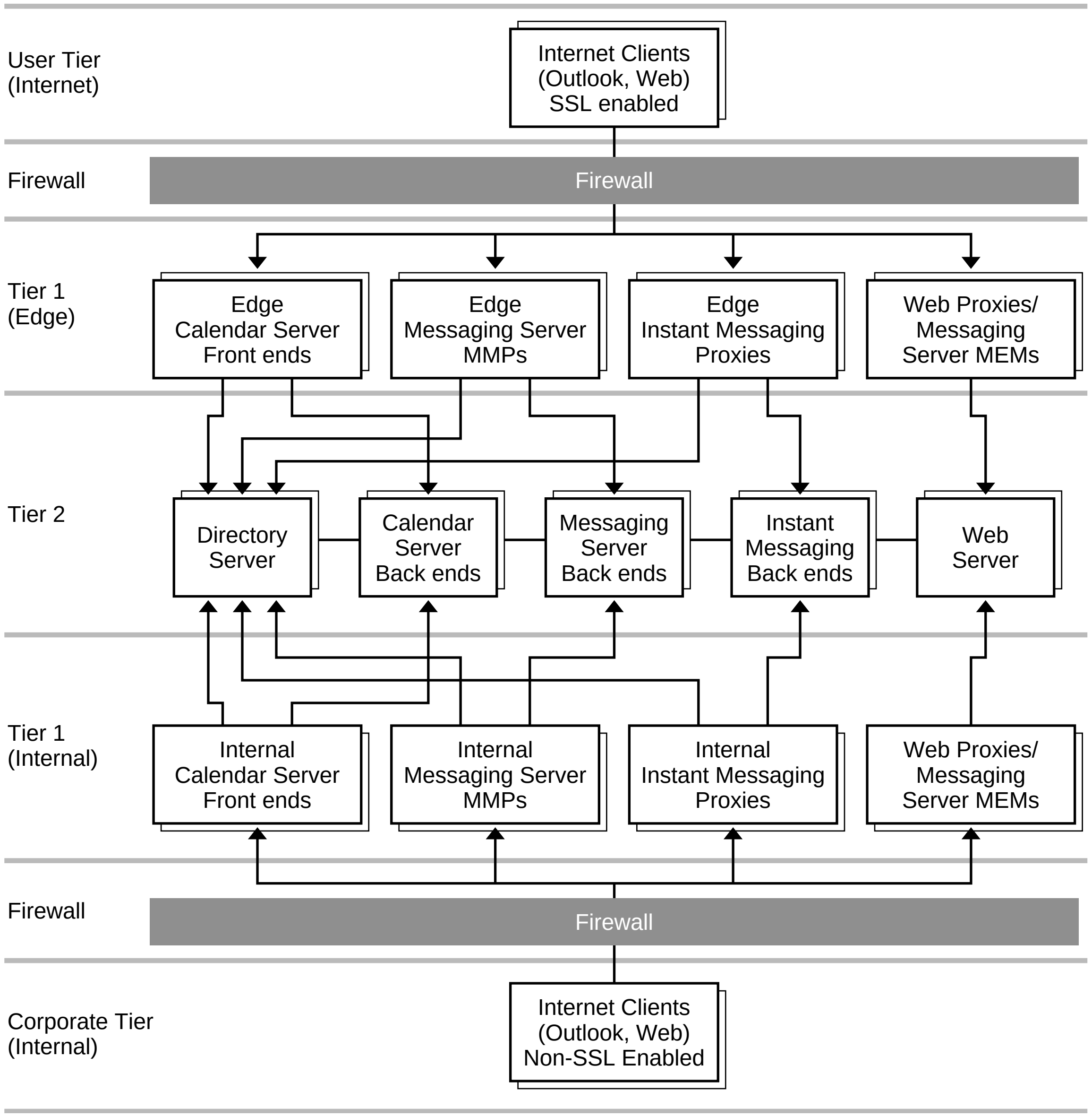
The edge logical architecture adds security for remote access to the two-tiered logical architecture. An edge deployment grants access to a remote, mobile workforce over the public Internet by using only name/password authentication (`SMTPAuth`). As messages travel to and from the corporate network over the public Internet, they are encrypted through the use of SSL. No virtual private network is involved. The internal side of the communications transmission is "in the clear" for maximum performance. Access is contained on the "edge" of the deployment, protecting the data stores from unauthorized intrusion.

Business reasons for an edge deployment include:

- Your workforce consists of mobile, remote workers.
- You do not want to install and maintain Communications Services servers at every remote site.

Figure 5-5 on page 89 represents the edge logical architecture.

Figure 5-5 Edge Architecture


User Tier (Internet)
Internet Clients (Outlook, Web) SSL enabled
Firewall
Firewall
Tier 1 (Edge)
Edge Calendar Server Front ends
Edge Messaging Server MMPs
Edge Instant Messaging Proxies
Web Proxies/ Messaging Server MEMs
Tier 2
Directory Server
Calendar Server Back ends
Messaging Server Back ends
Instant Messaging Back ends
Web Server
Tier 1 (Internal)
Internal Calendar Server Front ends
Internal Messaging Server MMPs
Internal Instant Messaging Proxies
Web Proxies/ Messaging Server MEMs
Firewall
Firewall
Corporate Tier (Internal)
Internet Clients (Outlook, Web) Non-SSL Enabled

In the preceding figure, the data stores are located in Tier 2, which is a secure, private network, connected only to the “edge” and “internal” front-end servers. Remote clients connect to front-end servers by using SSL. Internal clients do not need to use SSL to connect, as the assumption is made that internal access is inherently secure.

Edge Architecture Design Recommendations

- Capacity planning for the Edge tier is difficult to generalize. You should work with the hardware and software vendors who are supplying equipment for your deployment to develop a capacity plan. Nevertheless, you should implement the Realtime Blackhole List (RBL) at your site at the Edge tier. The RBL is a list of IP addresses whose owners refuse to stop the proliferation of spam.
- Design the Edge tier for minimal latency (less that one millisecond through the entire Edge tier).
- Use load balancing algorithms that are load-aware by CPU utilization or by the number of active connections. Round-robin is not an acceptable load-balancing model. With the exception of MTAs (stateless), use sticky-bit load balancing.
- Webmail clients need load balancers that can manage sticky bits, because Webmail interfaces do not share state across Webmail servers.

Benefits of a Single-tiered Architecture

The benefits of the single-tiered architecture come down to cost savings, as you do not have to purchase nor maintain additional hardware.

Answer the following questions to help decide if the single-tiered architecture is best for your enterprise:

- Is the threat of denial of service minimal?
- Is there no requirement for SSL?
- Do you have significant maintenance windows available?

Answering yes to these questions suggests that your enterprise could use a single-tiered architecture.

Benefits of a Two-tiered Architecture

All services within the Communications Services offering rely on network capabilities. A two-tiered architecture provides for a network design with two separate networks: the public (user-facing) network, and the private (data center) network.

Separating your network into two tiers provides the following benefits:

- **Hides Internal Networks.** By separating the public (user-facing) network and the private (data center) network, you provide security by hiding the data center information. This information includes network information, such as IP addresses and host names, as well as user data, such as mailboxes and calendar information.
- **Provides Redundancy of Network Services.** By provisioning service access across multiple front-end machines, you create redundancy for the system. By adding redundant messaging front-end servers, you improve service uptime by balancing SMTP requests to the available messaging front-end hosts.
- **Limits Available Data on Access Layer Hosts.** Should the access layer hosts be compromised, the attackers cannot get to critical data from the access hosts.
- **Offloads Tasks to the Access Layer.** By enabling the access layer to take complete ownership of a number of tasks, the number of user mailboxes on a message store increases. This is useful because the costs of both purchase and maintenance are much higher for store servers than for access layer machines (the second tier). Access layer machines are usually smaller, do not require large amounts of disk (see "MTA Performance Considerations" on page 185), and are rarely backed up. A partial list of features that are offloaded by the second tier includes:
  - Denial of Service protection
  - SSL
  - Reverse DNS
  - UBE (spam) and virus scanning
  - Initial authentication - Authentications to the Message Store should always succeed and the directory servers are more likely to have cached the entry recently.
  - LMTP - With support for LMTP between the MTA relays and the message stores, SMTP processing is offloaded and the need to do an additional write of the message into the MTA queues on the message stores is eliminated.

- **Simplifies End-user Settings in Client Applications.** By using a two-tiered architecture, end users do not have to remember the physical name of hosts that their messaging and calendar applications connect to. The Access-Layer Application hosts provide proxies to connect end users to their assigned messaging or calendar data center host. Services such as IMAP are connected to the back-end service using LDAP information to identify the name of the user's mailbox host. For calendar services, the calendar front-end hosts provide a calendar lookup using the directory server to create a back-end connection to the user's assigned calendar store host.

  This capability enables all end users to share the same host names for their client settings. For example, instead of remembering that their message store is `host-a`, the user simply uses the setting of `mail`. The MMP provides the proxy service to the user's assigned message store. You need to provide the DNS and load balancing settings to point all incoming connections for mail to one (or more) MMPs.

  By placing Calendar Server into two tiers, more than one Calendar Server back-end server can be used. Calendar Server's group scheduling engine enables users to schedule appointments with users whose calendars are on any of the back-end Calendar Server hosts.

  An additional benefit of this proxy capability provides geographically dispersed users to leverage the same client application settings regardless of their physical location. Should a user from Europe visit California, the user will be able to connect to the immediate access server in California. The user's LDAP information will tell the access server to create a separate connection on the user's behalf to the user's message store located in Europe.

  Lastly, this enables you to run a large environment without having to configure user browsers differently, simplifying user support. You can move a user's mailbox from one mail store to another without contacting the user or changing the desktop.

- **Reduces Network HTTP Traffic on the Data Center.** Both the Messaging Express Multiplexor (MEM) and the Calendar Server front-end greatly reduce HTTP traffic to the data center network. HTTP provides a connectionless service. For each HTML element, a separate HTTP request must be sent to the mail or calendar service. These requests can be for static data, such as an image, style sheets, JavaScript™ files, or HTML files. By placing these elements closer to the end user, you reduce network traffic on the back-end data center.

Horizontal Scalability Strategy

Scalability is critical to organizations needing to make the most cost-effective use of their computing resources, handle peak workloads, and grow their infrastructure as rapidly as their business grows. Keep these points in mind:

- How the system responds to increasing workloads: what performance it provides, and as the workload increases, whether it crashes or enables performance to gracefully degrade.
- How easy it is to add processors, CPUs, storage, and I/O resources to a system or network to serve increasing demands from users.
- Whether the same environment can support applications as they grow from low-end systems to mid-range servers and mainframe-class systems.

When deployed in a two-tiered architecture, the Communications Services offering is meant to scale very effectively in a horizontal manner. Each functional element can support increased load by adding additional machines to a given tier.

Scaling Front-end and Back-end Services

In practice, the method for scaling the front-end and back-end services differs slightly.

For Tier 1 elements, you start the scaling process when traffic to the front end grows beyond current capacity. You add relatively low cost machines to the tier and load balance across these machines. Thus, load balancers can precede each of the Tier 1 service functions as overall system load, service distribution, and scalability requirements dictate.

For Tier 2 elements, you start the scaling process when the back-end services have exceeded user or data capacity. As a general rule, design the Tier 2 services to accommodate just under double the load capacity of the Tier 1 services.

For example, for an architecture designed for 5,000 users, the Tier 1 front-end services are designed to support 5,000 users. The back-end services are then doubled, and designed to accommodate 10,000 users. If the system capacity exceeds 5,000 users, the front-end services can be horizontally scaled. If the overall capacity reaches 5,000 users, then the back-end services can be scaled to accommodate. Such design enables flexibility for growth, whether the growth is in terms of users or throughput.

Other Deployment Issues

This section describes some common Communications Services deployment best practices and other deployment considerations.

Implementing Local Message Transfer Protocol (LMTP) for Messaging Server

Best practices say you should implement LMTP to replace SMTP for message insertion. An LMTP architecture is more efficient for delivering to the back-end Message Store because it:

- Reduces the load on the stores. Relays are horizontally scalable while stores are not, thus it is a good practice to make the relays perform as much of the processing as possible.
- Reduces IOPS by as much as 30 percent by removing the MTA queues from the stores.
- Reduces the load on LDAP servers.

  The LDAP infrastructure is often the limiting factor in large messaging deployments.
- Reduces the number of message queues.

You need a two-tiered architecture to implement LMTP. See the *Sun Java System Messaging Server Administration Guide* for instructions on configuring LMTP:

http://docs.sun.com/doc/819-0105

Implementing Realtime Blackhole List (RBL)

The Mail Abuse Protection System's Realtime Blackhole List (MAPS RBL) is a dynamically updated list of known unsolicited bulk email (UBE) sources identified by source IP address. The Messaging Server SMTP server supports use of the RBL and can reject mail coming from sources identified by the RBL as originators of UBE or spam.

Implementing an RBL should be a consideration of every deployment. In general, a good RBL deployed in front of MTAs reduces traffic to the MTAs by a minimum of 10 percent, and in some cases, much higher.

RBLs, and anti-spam and anti-virus servers, such as BrightMail, can work together. For example, if your anti-spam server rejects 95 to 99 out of 100 emails from a particular IP address, you can add that IP address to the RBL. You can also adjust the RBL for BrightMail's false positives when you conduct your BrightMail analysis. Thus, you make the RBL much more proactive in handling a specific wave of UBE.

See the *Sun Java System Messaging Server Administration Reference* for information on configuring the `ENABLE_RBL` option of the MTA Dispatcher:

`http://docs.sun.com/doc/819-0106`

Using Logical Service Names

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

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

The following tables describes these logical entities.

Table 5-1 User Facing Logical Names

Example	Description
`mail.siroe.com`	Name of the server from which end users collect their email.
`imap.siroe.com`	Name of the IMAP server from which end users collect their email.
`pop.siroe.com`	Name of the POP server from which the end users collect their email.
`smtp.siroe.com`	Name of the SMTP server users set as outgoing mail server.
`webcal.siroe.com` or `ce.siroe.com`	Name of the Communications Express (formerly Calendar Express) server.

Table 5-2 Maintenance Level Logical Names

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

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

Maintenance Level	User Level
`relay-in.siroe.com`	N/A
`relay-out.siroe.com`	`smtp.siroe.com`
`mmp.siroe.com`	Any one or more of `mmp.siroe.com`, `pop.siroe.com`, and `imap.siroe.com`
`mem.siroe.com`	`webmail.siroe.com`
`storeAA.siroe.com` - `storeZZ.siroe.com`	N/A, hidden from end users
`calstore_aa.siroe.com` - `calstore_az.siroe.com`	N/A, hidden from end users

Chapter 6

Designing for Service Availability

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

This chapter contains the following sections:

High Availability Solutions Overview

High availability solutions for Communications Services vary from product to product.

For example, Messaging Server supports two different high availability solutions, Sun Cluster and Veritas Cluster Server (VCS). Messaging Server provides agents for each of these solutions.

Messaging Server and Calendar Server support different cluster topologies. Refer to the appropriate cluster product documentation for more information.

If you choose to use Application Server as a web container, you can take advantage of its high availability, load balancing, and cluster management capabilities.

Instant Messaging does not provide a Sun Cluster agent, nor does it support VCS. However, you can create a "more available" deployment by deploying redundant Instant Messaging multiplexors. In such a deployment, if one multiplexor fails, Instant Messaging clients are able to communicate to the back-end server through another available multiplexor.

In addition, you can build in availability to your Communications Services deployment by making infrastructure components, such as Directory Server, highly available.

The following sections in this chapter explain the options available for each component.

Automatic System Reconfiguration (ASR)

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

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

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

Directory Server and High Availability

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

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

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

The following table shows how you can design your directory for availability.

Table 6-1 Designing Directory Server for High Availability

Method	**Description**
Single-master replication	A server acting as a supplier copies a master replica directly to one or more consumer servers. In this configuration, all directory modifications are made to the master replica stored on the supplier, and the consumers contain read-only copies of the data.
Two-way, multi-master replication	In a multimaster environment between two suppliers that share responsibility for the same data, you create two replication agreements. Supplier A and Supplier B each hold a master replica of the same data and there are two replication agreements governing the replication flow of this multimaster configuration.

Table 6-1 Designing Directory Server for High Availability *(Continued)*

Method	Description
Four-way multi-master	Provides a pair of Directory Server masters, usually in two separate data centers. This configuration uses four-way MultiMaster Replication (MMR) for replication. Thanks to its four-way master failover configuration, this fully-connected topology provides a highly-available solution that guarantees data integrity. When used with hubs in the replication topology, load distribution is facilitated, and the four consumers in each data center allow this topology to scale for read (lookup) operations.
Sun Cluster Agent for Directory Server	Using Sun Cluster software provides the highest level of availability for your directory implementation. In the case of failure of an active Directory Server node, Sun Cluster provides for transparent failover of services to a backup node. However, the administrative (and hardware) costs of installing, configuring, and maintaining a cluster are typically higher than the Directory Server replication methods.

See the *Sun Java System Directory Service Deployment Planning Guide* for more information:

`http://docs.sun.com/doc/817-7607`

Application Server and High Availability

Because Communications Express is deployed in a web container (either Web Server or Application Server), consider making the web container highly available.

For example, Application Server Enterprise Edition enhances the core application server platform with high availability, load balancing and cluster management capabilities. The management capabilities of the Platform Edition are extended in Enterprise Edition to account for multi-instance and multimachine deployments.

Application Server's clustering support includes easy-to-configure groups of cloned application server instances to which client requests can be load balanced. Both external load balancers and load balancing web tier-based proxies are supported by this edition. Application Server EE provides failover for HTTP sessions and stateful session beans using the highly available database (HADB).

See the Application Server Enterprise Edition 8 2005Q1 documentation for more information:

`http://docs.sun.com/app/docs/coll/ApplicationServer8_ee_04q4`

Messaging Server and Calendar Server and High Availability

You can configure Messaging Server and Calendar Server to be highly available by using clustering software. Messaging Server supports both Sun Cluster and Veritas Cluster Server software. Calendar Server supports Sun Cluster software.

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

See "Key Concepts - Hardware Service Providers" in the *Sun Cluster Concepts Guide for Solaris OS* for more information on Sun Cluster topologies:

http://docs.sun.com/doc/817-6537

See "Configuring High Availability" in the *Sun Java System Messaging Server Administration Guide* for more information on configuring Messaging Server for high availability:

http://docs.sun.com/doc/819-0105

See "Setting Up a High Availability Configuration" in the *Sun Java System Calendar Server Administration Guide* for more information on configuring Calendar Server for high availability:

http://docs.sun.com/doc/819-0024

Instant Messaging and Availability

Instant Messaging does not provide a Sun Cluster agent, nor does it support Veritas Cluster Service. However, you can create a "more available" environment by deploying redundant Instant Messaging multiplexors, and by taking advantage of the Instant Messaging watchdog process.

Using Multiple Instant Messaging Multiplexors

In an Instant Messaging deployment of multiple multiplexors, if one multiplexor fails, Instant Messaging clients are able to communicate to the back-end server through another available multiplexor. Currently, you can only configure multiple multiplexors to speak to a single instance of Instant Messaging server. You cannot configure multiple multiplexors to talk to multiple instances of Instant Messaging.

Using the Instant Messaging Watchdog Process

Instant Messaging includes a watchdog process, which monitors and restarts the services that become unavailable for some reason (such as server lockup, crash, and so forth). In the event that you configure the watchdog process, and an Instant Messaging component stops functioning, the watchdog process shuts down then restarts the component.

Using Enabling Techniques and Technologies

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

Using Load Balancers

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

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

Using Directory Proxy Server

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

See the Directory Proxy Server documentation for more information:

`http://docs.sun.com/coll/DirectoryProxyServer_05q1`

Using Replica Role Promotion

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

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

See the *Sun Java System Directory Server Administration Guide* for more information:

`http://docs.sun.com/doc/817-7613`

Locating High Availability Product Reference Information

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

- **Sun Cluster**
  - *Sun Cluster Concepts Guide for Solaris OS*
    `http://docs.sun.com/doc/817-6537`
  - *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*
    `http://docs.sun.com/doc/817-6555`
  - *Sun Cluster Overview for Solaris OS*
    `http://docs.sun.com/doc/817-6536`
  - *Sun Cluster System Administration Guide for Solaris OS*
    `http://docs.sun.com/doc/817-6546`
- **Veritas Cluster Server**
  - *Veritas Cluster Server User's Guide*
    `http://ftp.support.veritas.com/pub/support/products/ClusterServer_UNIX/249745.pdf`

Understanding Remote Site Failover

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

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

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

Potential remote site failover solutions include:

- **Simple, less expensive scenario.** The remote site is not connected by large network bandwidth. Sufficient hardware is setup but not necessarily running. In fact, it might be used for some other purpose in the meantime. Backups from the primary site are shipped regularly to the remote site, but not necessarily restored. The expectation is that there will be some significant data loss and possibly a significant delay in getting old data back online. In the event of a failure at the primary site, the network change is manually started. Services are started, followed by beginning the `imsrestore` process. Lastly, the file system restore is started, after which services are brought up.
- **More complicated, more expensive solution.** Both Veritas and 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. If the Message Store servers are connected to storage arrays supporting remote replication, then the data copy to the remote site can be handled by the storage array controller itself. Storage arrays that offer a remote replication feature tend to be large, so the base cost of obtaining this solution is higher than using lower-end storage products.

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

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

Questions for Remote Site Failover

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

- What level of responsiveness does your site need?

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

NOTE Do not use both 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 7

Designing for Security

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:

For detailed product security information, see Chapter 13, “Planning Messaging Server Security” and Chapter 18, “Planning Calendar Server Security.”

Communications Services Security Overview

You manage security for a Communications Services deployment by taking a “defense in depth” approach. By individually securing the network, hardware platform, operating system, and applications themselves, you make each layer of the architecture secure. Security includes hardening each layer by closing unnecessary network ports and access mechanisms. You also minimize the number of installed software packages so that only those packages required by the system are available. Finally, you secure and insulate the layers from unintended access within the network.

You can implement a Messaging Server proxy server to augment data security. A proxy server placed on the firewall with the Messaging Server behind it prevents attacks on the information on the Messaging Server.

Calendar Server provides a number of security levels to protect users against eavesdropping, unsanctioned usage, or external attack. The basic level of security is through authentication. Calendar Server uses LDAP authentication by default, but also supports the use of an authentication plugin for cases where an alternate means of authentication is desired. Integration with Access Manager enables Calendar Server to take advantage of its single sign-on capability.

Instant Messaging ensures the integrity of communications through its multiple authentication mechanisms and secure SSL connections. Integration with Portal Server and Access Manager bring additional security features, services-based provisioning access policy, user management, and secure remote access.

NOTE To ensure a completely secure environment, the deployment needs a time server to synchronize the internal clocks of the hosts being secured.

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.

The five steps to developing a security strategy, as listed in RFC 2196, the *Site Security Handbook*, include:

1. Identifying 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. Determining what you are trying to protect it from.

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

3. Estimating 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. Implementing 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 reviewing 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*:

   http://docs.sun.com/doc/819-0105

Your security strategy should also plan for:

- Physical Security
- Server Security
- Operating System Security
- Network Security
- Messaging Security
- Application 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.

Operating System Security

Reduce potential risk of security breaches in the operating environment by performing the following, often termed "system hardening:"

- **Minimize the size of the operating environment installation.** When installing a Sun server in an environment that is exposed to the Internet, or any untrusted network, reduce the Solaris software installation to the minimum number of packages necessary to support the applications to be hosted. Achieving minimization in services, libraries, and applications helps increase security by reducing the number of subsystems that must be maintained.

  The Solaris Security Toolkit provides a flexible and extensible mechanism to minimize, harden, and secure Solaris systems. The primary goal behind the development of this toolkit is to simplify and automate the process of securing Solaris systems. For more information see:

  `http://wwws.sun.com/software/security/jass/`

- **Track and monitor file system changes.** Within systems that require inclusion of security, a file change control and audit tool is indispensable as it tracks changes in files and detects possible intrusion. You can use a product such as Tripwire for Servers, or Solaris Fingerprint Database (available from SunSolve Online).

Network Security

The recommended deployment configuration, to support both horizontal scalability and service security, is to place the access layer of the architecture behind a firewall. In a two-tiered architecture, use two firewalls, creating a DMZ. This enables access to the information delivery elements, the calendar and messaging front ends, while protecting the main service elements on the internal network behind a second firewall. Such a configuration also enables the access layer and data layer elements to be scaled independently, accommodating traffic and storage elements.

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 application servers 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.

- Understanding Security Misconceptions

  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).

See Chapter 13, "Planning Messaging Server Security" for more information.

Application Security

The Communications Services product portfolio provides features that ensure the security and integrity of business communications. Communications Services offer extensive "built-in" security features, such as:

- Authentication
- Message and session encryption
- Virus and spam protection
- Archiving and auditing of communications
- End-user configurable privacy options

Implementing Secure Connections

Communications Services support security standards such as SSL/TLS, S/MIME, and SAML. SSL/TLS enables all communication between clients and servers to take place inside an encrypted session. Through integration with Portal Server, additional authentication mechanisms are available out-of-the-box, as well as single sign-on capabilities across the applications.

NOTE There is no SSL support between the Communications Express application in the web server and the Calendar Server `cshttpd` daemon.

If you want to implement public key data security, you must select mail clients that support public key infrastructure and key choice. Communications Services products are capable, out of the box, of participating in the transmission and storage of messages so encrypted. Secure/Multipurpose Internet Mail Extension (S/MIME) is available on the Communications Express Mail client. Communications Express Mail users who are set up to use S/MIME can exchange signed or encrypted messages with other users of Communications Express Mail, Microsoft Outlook Express, and Mozilla mail systems.

NOTE The Webmail client from previous versions of Messaging Server is not capable of generating or decoding encrypted messages.

A more commonly used mechanism for data security protects the data across the wire only (that is, from client to server) by using SSL encryption on the connections used to transmit data between various messaging agents. This solution is not as complete as public key encryption, but is far easier to implement and is supported by many more products and service providers.

What problem does using SSL from client to server solve? An organization assumes that it controls its own corporate network and that data transmitted on that network is safe from non-employees. Mail sent to anyone from outside the corporate network using the corporation's infrastructure transmits the data over an encrypted connection to the corporation's network. Likewise, all mail picked up by a corporate user from outside the corporate network will be transmitted over an encrypted connection. Thus, if the enterprise's assumption about the safety of the internal network is true, and its employees use only sanctioned servers for transmission between themselves and other employees, mail between employees is safe from external attack.

What problem doesn't this solution solve? First of all, this approach does not protect the data from unintended viewing by non-intended recipients of the data who have access to the organization's internal network. Secondly, there is no protection offered for data being transmitted between employees and their external partners, customers, or suppliers. The data travels across the public Internet in a completely insecure fashion.

However, this problem can be remedied by configuring SSL encryption between MTA routers at both the enterprise's and customer's network. This type of solution requires setup for each private connection you want to use. In so doing, you add an important additional layer of security with customer or partner data being sent or received via email. Using MTAs and SSL, companies can save money by using the public Internet as the transport, but force the MTAs to use SSL for their partners. This solution does not take into account other network traffic to and from partners. Nevertheless, mail is usually a large proportion of the traffic, and because companies can pay based on data transmitted, using the public Internet is usually cheaper.

Implementing Secure Connections Using Two Different Certificate Authorities (CAs)

You can implement SSL connections between server and clients, for example, from Messaging Server, and to other servers in your deployment as well (Web Server, Calendar Server, Directory Server). If desired, you can use two different Certificate Authorities (CAs), one for the server and one for the client.

In such a scenario, you can use one CA to issue server certificates, and another CA to issue client certificates. If you want the client to accept the server's certificate as genuine, you will need to load the CA certificate for the server into the client's certificate DB. If you want the server to accept the client as genuine, you will need to load the CA certificate for the client into the server's certificate DB.

Understanding Security Misconceptions

This section describes common 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 Communications Services deployment, review the Computer Emergency Response Team (CERT) Coordination Center site:

http://www.cert.org

Chapter 8

Understanding Schema and Provisioning Options

This chapter describes the schema and provisioning options for Communications Services. Because of the complexity in provisioning Communications Services, you need to understand your options before installing the product.

This chapter contains the following sections:

- Understanding Schema Choices
- Understanding Provisioning Tools

Understanding Schema Choices

This section describes the schema options that are available and supported with Communications Services, and how to decide which to use.

Understanding Messaging Server Schema Choices

Two schema options are available and supported with Messaging Server: Sun Java System LDAP Schema version 1 and Sun Java System LDAP Schema version 2.

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 for Messaging Server

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

- Are you integrating Messaging Server with other Java Enterprise System component products, such as Portal Server or Access Manager, which provide 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.

LDAP Schema 1 and Messaging Server

LDAP Schema 1 is a provisioning schema that consists of both an Organization Tree and a DC Tree. This set of schema (at the time, it was simply called "schema") was supported in previous Messaging Server 5.x versions.

In Schema 1, when Messaging Server searches for user or group entries, it looks at the user's or group's domain node in the DC Tree and extracts the value of the `inetDomainBaseDN` attribute. This attribute holds a DN reference to the organization subtree containing the actual user or group entry.

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 and Messaging Server Supported Provisioning Tools

Schema 1 supports Sun™ ONE Delegated Administrator for Messaging (formerly called iPlanet Delegated Administrator) as well as LDAP provisioning tools. For more information, see "Understanding Provisioning Tools" on page 125.

LDAP Schema 2 (Native Mode) and Messaging Server

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

The native mode uses search templates to search the 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 Communications Services 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 Communications Services 5.x installation that uses Schema 1, and you want to integrate Communications Services with other Java Enterprise Server products, you should migrate your directory to Schema 2 after you upgrade to Communications Services 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:

`http://docs.sun.com/doc/819-0112`

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 and Messaging Server Supported Provisioning Tools

Schema 2 supports Sun Java System Communications Services Delegated Administrator. For more information, see "Understanding Provisioning Tools" on page 125.

LDAP Schema 2 Compatibility Mode and Messaging Server

Schema 2 compatibility mode is an interim mode between Schema 1 and Schema 2 native mode. Schema 2 compatibility mode supports both schemas and enables you to retain the existing two-tree design you already have. Schema 2 compatibility mode also assumes that you have installed Access Manager 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, Access Manager, 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:

`http://docs.sun.com/doc/819-0112`

Understanding Calendar Server Schema Choices

Two schema options are available and supported with Calendar Server: Sun Java System LDAP Schema version 1 and Sun Java System LDAP Schema version 2.

NOTE Refer to 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 for Calendar Server

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

- Are you integrating Calendar Server with other Java Enterprise System component products, such as Portal Server or Access Manager, which provide single sign-on capabilities?

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

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

  If you are installing Calendar Server for the first time, use Schema 2 Native Mode.

  If you are upgrading from an older version of Calendar Server, you can either use Schema 1 or Schema 2 Native or Compatibility Mode.

- Do you plan to use either Access Manager CLI utilities for provisioning or single sign-on?

  If you answer Yes, use Schema 2 Native or Compatibility Mode.

- Do you want to use the Calendar Server `csdomain` utility for provisioning domains?

  If you answer Yes, use Schema 2 Native or Compatibility Mode. If you don't plan to use the `csdomain` utility, and you have an existing Calendar Server installation, use Schema 1.

- If you don't want to use either Access Manager or Calendar Server CLI utilities for provisioning, use can use either Schema 2 Native Mode for new installations, or Schema 1 or Schema 2 Compatibility Mode for existing Calendar Server installations.

LDAP Schema 1 and Calendar Server

LDAP Schema 1 is a provisioning schema that consists of both an Organization Tree and a DC Tree. This set of schema (at the time, it was simply called "schema") was supported in previous Calendar Server 5.x versions.

When Calendar 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 Calendar Server should use Schema 1.

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

LDAP Schema 1 and Calendar Server Supported Provisioning Tools

Schema 1 supports LDAP provisioning tools. For more information, see "Understanding Provisioning Tools" on page 125.

LDAP Schema 2 (Native Mode) and Calendar Server

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

The native mode uses search templates to search the 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 Communications Services 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 Communications Services 5.x installation that uses Schema 1, and you want to integrate Communications Services with other Java Enterprise Server products, you should migrate your directory to Schema 2 after you upgrade to Communications Services 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:

```
http://docs.sun.com/doc/819-0112
```

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 and Calendar Server Supported Provisioning Tools

Schema 2 supports Sun Java System Communications Services Delegated Administrator. For more information, see "Understanding Provisioning Tools" on page 125.

LDAP Schema 2 Compatibility Mode and Calendar Server

Schema 2 compatibility mode is an interim mode between Schema 1 and Schema 2 native mode. Schema 2 compatibility mode supports both schemas and enables you to retain the existing two-tree design you already have. Schema 2 compatibility mode also assumes that you have installed Access Manager 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, Access Manager, 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:

`http://docs.sun.com/doc/819-0112`

Understanding Provisioning Tools

This section describes supported provisioning tools that enable you to query, modify, add, or delete user, group, and domain entry information in your LDAP directory.

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 for Messaging Server" on page 120, you should use Table 8-1 on page 127 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
- LDAP Provisioning Tools for Messaging Server
- Delegated Administrator and Messaging Server
- Comparing Messaging Server Provisioning Tool Options

Sun ONE Delegated Administrator for Messaging

Sun ONE Delegated Administrator for Messaging (formerly called iPlanet Delegated Administrator) 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 for Messaging Server

Schema 1 users and groups can be provisioned using the LDAP Directory tools (Schema 2 is not supported). Unlike the Delegated Administrator graphical and command-line interfaces, you can directly provision users and groups by adding, removing, and modifying the LDIF records through LDAP without having to use a user interface.

Delegated Administrator and Messaging Server

Access Manager 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 Delegated Administrator. 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.

See the *Sun Java System Communications Services Delegated Administrator Guide* for installation details:

`http://docs.sun.com/doc/819-0114`

Comparing Messaging Server Provisioning Tool Options

The following table shows the various supported schema, provisioning tools, provisioning limitations, and recommended documentation for additional information.

Table 8-1 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 8-1 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.
Delegated Administrator Uses: Schema 2	Provides graphical and command-line interfaces 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 Access Manager • Sun Java System Access Manager must be installed to enable this command-line interface.	Read the *Sun Java System Communications Services Delegated Administrator Guide*. Provides syntax and usage for the command-line utility.

Understanding Calendar Server Provisioning Tools

Through supported Calendar Server provisioning tools, you can query, modify, add, or delete user, group, and domain entry information in your LDAP directory. This section examines these Calendar Server provisioning tools.

In addition to the questions asked in “Deciding Which Schema to Use for Calendar Server” on page 123, you should use Table 8-2 on page 130 to evaluate your schema and provisioning tool options.

NOTE Prior to installing and configuring Calendar Server, you need to decide upon a schema model and tool or tools for provisioning your Calendar Server entries.

The following sections provide high-level information about the supported provisioning tools:

- LDAP Provisioning Tools for Messaging Server
- Delegated Administrator and Calendar Server
- Comparing Messaging Server Provisioning Tool Options

LDAP Provisioning Tools for Calendar Server

Schema 1 users and groups can be provisioned using the LDAP Directory tools (Schema 2 is not supported). You can directly provision users and groups by adding, removing, and modifying the LDIF records through LDAP without having to use a user interface.

Delegated Administrator and Calendar Server

Access Manager 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 Delegated Administrator utility (command-line interface). 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 Calendar Server.

NOTE Even though you install Access Manager, there is no graphical user interface compatibility with Calendar Server. Therefore, to provision Calendar Server users and groups with an interface, you can only use the Delegated Administrator command-line interface.

See the *Sun Java System Communications Services Delegated Administrator Guide* for installation details:

http://docs.sun.com/doc/819-0114

Comparing Calendar Server Provisioning Tool Options

The following table shows the various supported schema, provisioning tools, provisioning limitations, and recommended documentation for additional information.

Table 8-2 Calendar Server Provisioning Mechanisms

Supported Provisioning Tool	Provisioning Tool Functionality	Provisioning Tool Limitations	For Further Information
LDAP Provisioning Tools Uses: Schema 1	Provides tools to directly modify LDAP entries or for creating custom provisioning tools.	Incompatible with Sun Schema 2 and with other Java Enterprise System products.	Read the *Sun ONE Calendar 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.

Table 8-2 Calendar Server Provisioning Mechanisms *(Continued)*

Supported Provisioning Tool	Provisioning Tool Functionality	Provisioning Tool Limitations	For Further Information
Delegated Administrator 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. Note: Currently, there is no graphical interface for managing Calendar Server provisioning. You must use the command-line interface.	• Not backwardly compatible with Sun Schema 1. • No GUI provisioning tool to use with Sun Java System Access Manager • Sun Java System Access Manager must be installed to enable this command-line interface.	Read the *Sun Java System Communications Services Delegated Administrator Guide.* Provides syntax and usage for the command-line utility.

Part II

Deploying Messaging Server

Chapter 9

Introduction to Messaging Server Software

Sun Java System Messaging Server is a powerful 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 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:

- What Is a Messaging System?
- Messaging Server Support for Standards and Functionality
- Messaging Server Software Architecture

What Is a Messaging System?

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

- Accept incoming mail from external sites

- Determine the user mailbox to deliver these messages to and route them accordingly
- Accept incoming mail from internal hosts
- Determine the destination system to deliver these messages to and route them accordingly

Central to an email system architecture is the messaging server itself, 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. The DNS server must be in place before deploying your email system.

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

- Load balancing
- Firewalls
- High availability

See Chapter 11, "Developing a Messaging Server Architecture" for more information on these topics.

Messaging Server Support for Standards and Functionality

This section describes standards supported by Messaging Server, as well as other supported functionality.

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/819-0106`

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.

When you define a domain, there must be a corresponding domain entry in the directory, that is, you must have an LDAP entry for the domain. Attributes such as `mailAlternateAddress` and `mailEquivalentAddress` depend on the existence of domain entries in the directory. Contrast this with vanity domains, which are domain names associated with an individual user, not with a specific server or hosted domain. The vanity domain does not have an LDAP entry for the domain name.

NOTE Because of their increased operational overhead, vanity domains are not recommended.

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 LDAP Schema version 1 (Schema 1) and Sun Java System LDAP Schema version 2 (Schema 2). The provisioning options will depend on which schema you have chosen. See Chapter 15, "Understanding Messaging Server Pre-Installation Considerations and Procedures" for more information.

Messaging Server provisioning for Schema 2 is done using Delegated Administrator, as documented in the *Sun Java System Communications Services Delegated Administrator Guide*:

`http://docs.sun.com/doc/819-0114`

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 Communications Services:

  `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 Access Manager console provides minimal Messaging Server and Calendar Server LDAP user entry provisioning using Access Manager 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.

Delegated Administrator, which is described in the *Sun Java System Communications Services Delegated Administrator Guide*, is the recommended mechanism for provisioning Communications Services users:

`http://docs.sun.com/doc/819-0114`

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, video, and other forms of communication.

Support for Webmail

Messaging Server currently supports two client user interfaces (UI):

- Messenger Express
- Communications Express

Going forward, no new features will be added to the Messenger Express user interface. It has been deprecated in favor of the Communications Express user interface. Sun Microsystems, Inc. will announce an end-of-life timeline for Messenger Express at a future date.

See the Communications Express documentation for more information:

http://docs.sun.com/coll/MessagingServer_05q1

Messaging Server 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
- Client IP address access filters to POP, IMAP, SMTP, and HTTP
- Filtering of unsolicited bulk email using system-wide, per-user, and server-side Sieve rules

Messaging Server Administration 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.

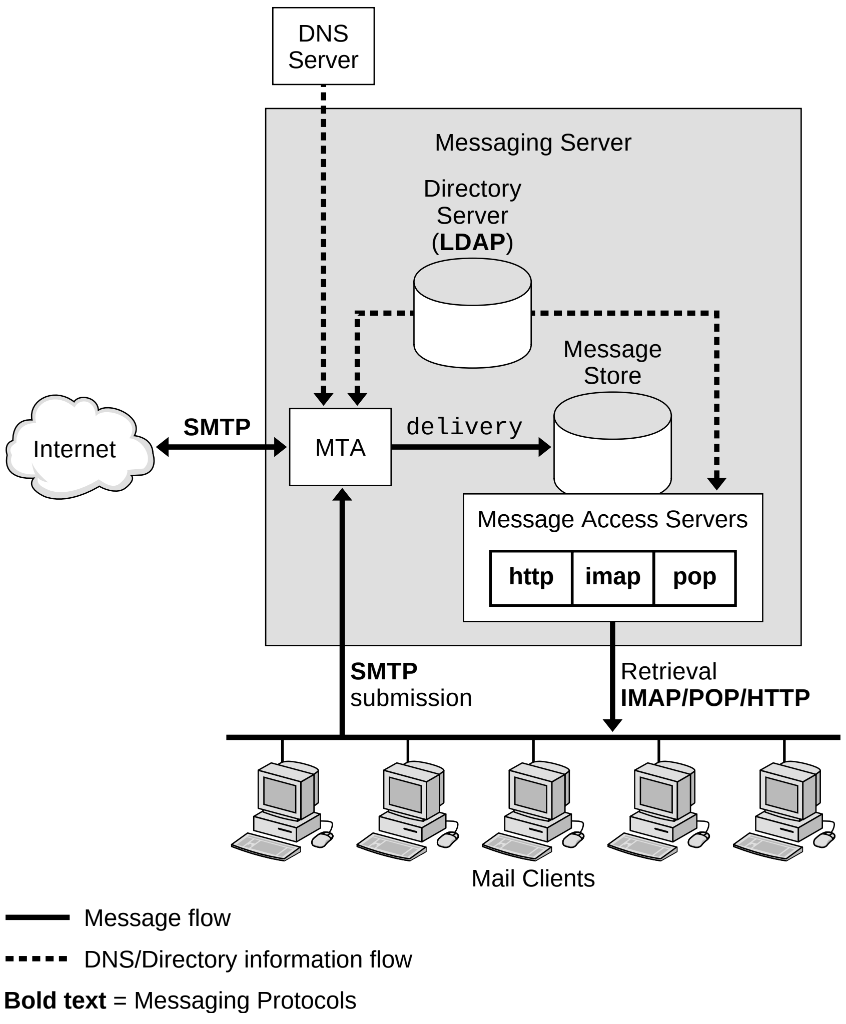
See the *Sun Java System Messaging Server Administration Guide* for more information:

`http://docs.sun.com/doc/819-0105`

Messaging Server Software Architecture

Figure 9-1 on page 141 shows a simplified standalone view of Messaging Server. While this particular deployment is not recommended for large deployment because it does not scale well, it does illustrate the individual components of the server.

Figure 9-1 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 delivers messages to a local 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 manipulate messages on the server.
- **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 also stores passwords, and any other information needed by the MTA or Message Store to authenticate users.

  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.
- **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), 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.

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 Communications Services Delegated Administrator utility or by modifying the directory using LDAP.

Messaging Server components can be administered by the Administration Server console. In addition, Messaging Server provides a set of command-line interfaces and configuration files. 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.

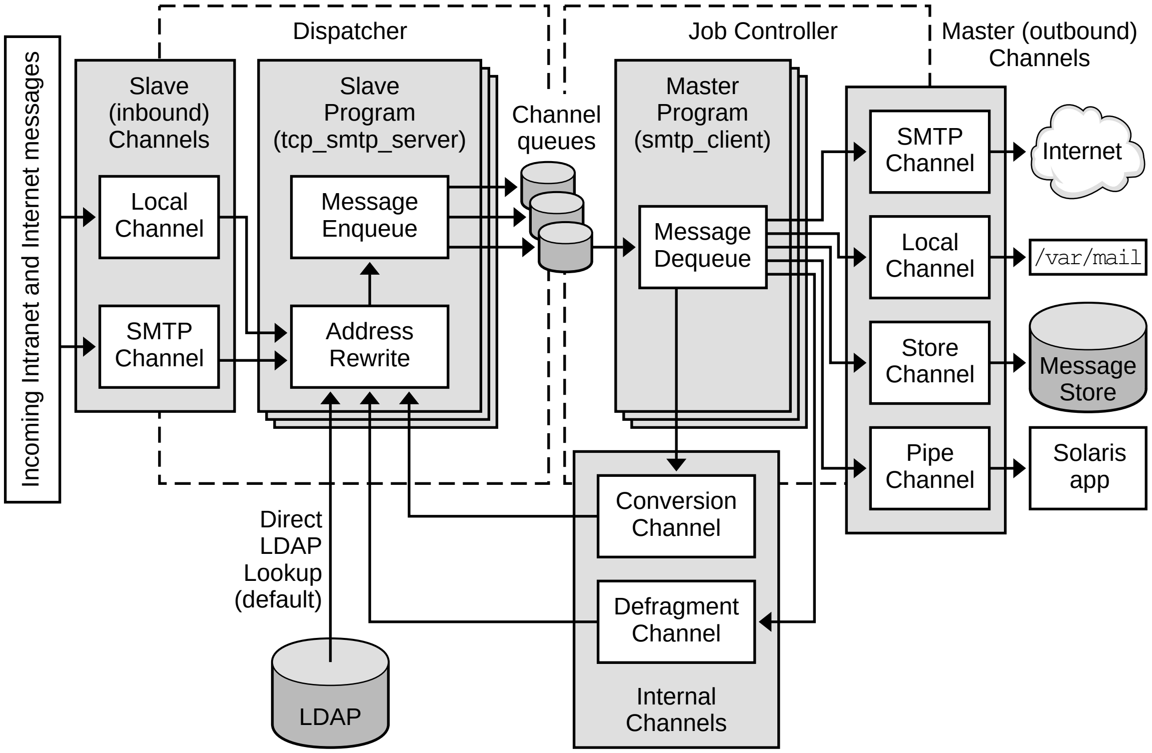
The Message Transfer Agent (MTA)

The MTA routes, transfers, and delivers Internet mail messages for Messaging Server. Mail flows through interfaces known as *channels*. Each channel consists of one or a pair of agent programs and a set of configuration information. The agent programs are a *slave program*, which handles mail coming into the channel, and a *master program*, which handles mail as it leaves the channel. There is a message queue for storing messages that are destined to be sent to one or more of the interfaces associated with any channel. 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 in a two-tiered configuration. These channels communicate with the Message Store on another system over LMTP instead of SMTP. Both master and slave channels are provided.
- **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. A master channel is provided.
- **Local Channel.** Delivers mail to `/var/mail`. Provides for compatibility with older UNIX mail clients. A master channel is provided.
- **Reprocessing Channel.** Useful for messages that are resubmitted. A master channel is provided.
- **Defragmentation Channel.** Reassembles partial messages into the original complete message to support the MIME message/partial content type. A master channel is provided.
- **Conversion Channel.** Performs body part by body part conversion on messages. Useful for rewriting addresses or re-formatting messages. A master channel is provided.
- **Message Store Channel.** Provides for local delivery to the message store.

The following figure illustrates the process. You can configure channels individually and direct mail to specific channels based on the address.

Figure 9-2 Channel Architecture



Channel programs perform one of two functions:

- SMTP slave programs enqueue messages into message queues for further processing by the MTA, or reject the message so it is not accepted onto the system, and accept messages from other interfaces.
- Master programs process the message from its queue area, and enqueue it on the same system for further processing by another channel, or transmit messages off system to other interfaces, deleting them from their queue after they are sent, or 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 destination, 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 more information on MTA concepts:

http://docs.sun.com/doc/819-0105

Direct LDAP Lookup

The MTA looks up the information directly from the LDAP server. The direct lookup provides a scalable, fast, and 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.

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/queued. 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 whose channel host name is `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*:

http://docs.sun.com/doc/819-0105

The Job Controller

Master channel programs are run under the control of the job controller, a program that controls the message queues and invokes the channel programs to do the actual message delivery. The job controller is 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.

Job controller configuration settings 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 to do currently.

Slave channels, which accept external messages, by queueing a message, 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 in that queue. This job creation might not be necessary if the job controller determines that an existing channel job can process the newly queued 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.

NOTE There are currently only two slave channels, SMTP slave and LMTP slave. These programs are controlled by the dispatcher, which is described next.

Dispatcher

The dispatcher is another process that is always present on a Messaging Server system. It is a multithreaded traffic dispatcher, which dispatches incoming SMTP or LMTP connections to the pool of SMTP or LMTP server threads for protocol-specific processing. The SMTP and LMTP servers programs provide a pool of worker threads at the disposal of the dispatcher. After it processes a message by either rejecting the message or enqueuing it into its destination channel, the worker thread is ready to accept more work from the dispatcher.

The dispatcher can block incoming traffic based on IP address and throttles it to prevent denial of service attacks. It also creates and shuts down SMTP or LMTP server processes based on load and configuration. Therefore the SMTP or LMTP slave channel programs are under the control of the dispatcher, not the job controller.

Local Mail Transfer Protocol (LMTP)

As of the Messaging Server 6.0 release, you can configure LMTP for delivery to the Message Store in a multi-tiered 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:

`http://docs.sun.com/doc/819-0105`

NOTE By design, LMTP is intended for use in multi-tier deployments. It is not possible to use LMTP with single-system deployments. Also, the Messaging Server's LMTP service as implemented is not designed to work with other LMTP servers or other LMTP clients.

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 HTTP server (`mshttpd`) to provide messaging capabilities to Communications Express in a web browser. 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 IMAP or Webmail 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. Messaging Server supports sharing folders across multiple stores by using the IMAP protocol.

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 file. 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 for user files. 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 to store new users.

NOTE If the partition gets full, users on the partition will not be able to store additional messages. There are several ways to address the problem:

- Reduce the size of user mailboxes
- If you are using volume management software, add additional disks
- Create additional partitions and move mailboxes to the new partitions

For more information, see the “Managing the Message Store” chapter in the *Sun Java System Messaging Server Administration Guide*:

`http://docs.sun.com/doc/819-0105`

The 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 a database format for faster access. The information in the system area can be reconstructed from the user area. Messaging Server contains a database snapshot function. When needed, you can quickly recover the database to a known state. Messaging Server also has 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 per-user quotas. 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.

Messaging Server and 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 Access Manager is not used is different from the one used by Access Manager. Messaging Server supports both Sun Java System LDAP Schema 1 and 2, and allows for transition and migration of the schemas.

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:

`http://docs.sun.com/db/coll/DirectoryServer_05q1`

Provisioning Messaging Users

See Chapter 8, “Understanding Schema and Provisioning Options” for information on schema and provisioning options for Messaging Server users.

Chapter 10

Planning a Messaging Server 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 Server deployment to enable you to obtain the right sizing data by which you can make deployment decisions. It also provides the context and rationale for the Messaging Server sizing process.

The chapter contains the following sections:

- Collecting Messaging Server Sizing Data
- Using a Messaging Server Load Simulator
- Assessing Your Messaging Server System Performance
- Developing Messaging Server Architectural Strategies

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.

Collecting Messaging Server Sizing Data

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

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

Determining Messaging Peak Volume

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

Analyzing peak volume involves three basic operations:

1. Determining when and for how long the peaks occur
2. Sizing your deployment against peak volume load assumptions

   Once patterns are analyzed, choices can be made to help the system handle the load and provide the services that users demand.
3. Making sure that your Messaging Server deployment can support the peak volume that you have determined.

Creating Your Messaging Usage Profile

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

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

To create a usage profile, answer the following questions:

1. What is the number of users on your system?

   When counting the number of users on your system, account for not only the users who have mail accounts and can log in to the mail system, but also the users with mail accounts who are currently not logged onto the system. In particular, note the difference between active and inactive users in Table 10-1 on page 155.

   Table 10-1 Active Versus Inactive Messaging 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 in this discussion 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 Sun Client 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 10-2 on page 156 defines these terms.

Table 10-2 Messaging 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 Messenger Express or for IMAP users. 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. Messages in trash or wastebasket folders still take up disk space and quota.

5. How many messages enter your messaging system from the Internet?

   The number of messages should be measured in messages per second during your peak volume.

6. How many messages are sent by your users to:

   - End users on your mail system?
   - The Internet?

   This number of messages is also measured in messages per second during the peak volume.

7. What is the distribution of messages in different size ranges?

   For example:

   - Less than 5 Kbytes?
   - Between 5 Kbytes - 10 Kbytes?
   - Between 10 Kbytes -100 Kbytes?
   - Between 100 Kbytes - 500 Kbytes?
   - Between 500 Kbytes -10 MB?
   - Greater than 10 MB?

   If the distribution of message sizes is not available, use the average message size on your mail system, however it is not as effective as size ranges.

   The size of messages is particularly important, because it affects the rate of delivery of the MTA, the rate of delivery into the Message Store, the rate of message retrieval, and processing by anti-virus or anti-spam filters.

8. Will you be using SSL/TLS? If yes, what percentage of users and what type of users?

   For example, in a particular organization, 20 percent of IMAP connections during peak hours will enable SSL.

9. Do you plan on using any SSL crypto accelerator hardware?

10. Will you be using virus scanning or other specialized message processing and will this processing be enabled for all users?

    Depending on your Messaging Server configuration, the MTA will need to scan all messages to match criteria specified in specialized processing, thus increasing load on the system.

11. For POP users, will you have a policy restricting how often they can access mail? If so, how often?

12. For IMAP users, will you enforce a standard client or allow users to choose their own?

    Different IMAP clients make different numbers of concurrent connections to the server. Thus, a power user with many open folders might have many concurrent connections.

13. Will you allow users to share folders? If so, will you allow all users or only some?

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

Additional Questions

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

1. How much redundancy do you want in your deployment?

   For example, you might consider high availability. Consider how much down time is allowed, and if you need clustering technology.

2. What backup and restore strategy do you have in place (such as disaster recovery, mailbox restores, and site failover)? What are the expected times to accomplish recovery tasks?

3. Do you need a DMZ to separate your internal and external networks? Are all users using the internal network? Or do some of them connect by using the Internet?

   You might need proxy servers (MMP, MEM) and separate MTA layers.

4. What are your response time requirements? What are your throughput requirements?

5. What is your specific criteria for resource utilization? Can your CPUs be 80 percent busy on average? Or only at peak?

6. Will you have messaging servers at different geographic locations? Do you expect users' mail to be located geographically?

7. Do you have archiving requirements for keeping mail messages for a certain length of time?

8. Do you have legal requirements to log all messages? Do you need to keep a copy of every message sent and received?

Defining Your Messaging User Base

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

- Lightweight POP
- Heavyweight POP
- Lightweight IMAP
- Mediumweight IMAP
- Mediumweight Messenger Express/Communications 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 Messaging Server Load Simulator" on page 161).

Lightweight POP

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

Heavyweight POP

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

Lightweight IMAP

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

Mediumweight IMAP

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

Mediumweight Messenger Express/Communications Express

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

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

Using a Messaging Server Load Simulator

To measure the performance of your Messaging Server, use your messaging user bases (described in "Defining Your Messaging User Base" on page 159) and your messaging usage profile (described in "Creating Your Messaging Usage Profile" on page 154) 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 Client Services for recommended load simulators and support.

Assessing Your Messaging Server System Performance

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

Messaging Server Memory Utilization

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

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

Messaging Server 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).

Messaging Server Disk Capacity

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

For the Message Store partitions, the storage requirement is the total size of all messages plus 30 percent overhead.

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

NOTE Your deployment planning needs to include how you want to back up the Message Store for disaster recovery. Messaging Server supports Solstice Backup (Legato Networker), the `imsbackup` utility, and file system snapshot backup. You might want to store your backup media remotely. The more frequently you perform a backup, the better, as long as it doesn't impact server operations.

Messaging Server Network Throughput

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

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

- To avoid bottlenecks, ensure that the network infrastructure can handle the load.
- Partition your network. For example, use 100 Mbps Ethernet for client access and 1 GB Ethernet for the backbone.
- To ensure that sufficient capacity exists for future expansion, 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.

Messaging Server 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 Messaging Server Architectural Strategies

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

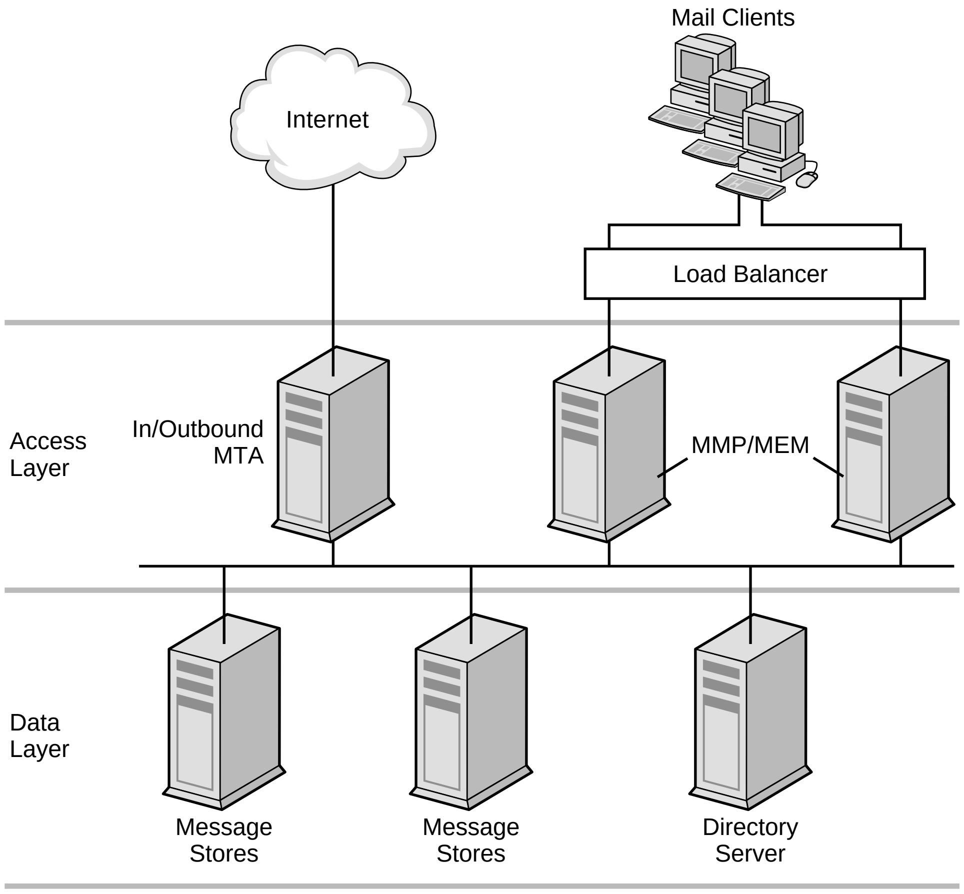
The following sections point out sizing considerations when you deploy two-tiered and one-tiered architectures.

NOTE For detailed information on planning your architecture, see Chapter 11, "Developing a Messaging Server Architecture."

Two-tiered Messaging Server Architecture

A two-tiered architecture splits the Messaging Server deployment into two layers: an access layer and a data layer. In a simplified two-tiered deployment, you might add an MMP and an MTA to the access layer. The MMP acts as a proxy for POP and IMAP mail readers, and the MTA relays transmitted mail. The data layer holds the Message Store and Directory Server. Figure 10-1 on page 165 shows a simplified two-tiered architecture.

Figure 10-1 Simplified Messaging Server Two-tiered Architecture



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

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

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

➤ To Size the Message Store

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

1. Determine the number of store machines and concurrent connections per machine based on the figures you gather by using a load simulator. For more information on sizing tools, see "Using a Messaging Server Load Simulator" on page 161.
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 Client Services is often asked to specify a recommendation for the maximum number of users on a message store. Such a recommendation cannot be given without understanding:

- Usage patterns (as described in "Using a Messaging Server Load Simulator" on page 161).
- The maximum number of active users on any given piece of hardware within the deployment.
- Backup, restore, and recovery times. These times increase as the size of a message store increases.

➤ To Size Inbound and Outbound MTAs

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

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

1. From the raw performance of the inbound MTA, add SSL, virus scanning processes, and other extraordinary message processing.
2. Account for denial of service attacks at peak volume in the day.
3. Add enough MTAs for load balancing and for redundancy as appropriate.

   With redundancy, one or more of each type of machine can still handle peak load without a substantial impact to throughput or response time.

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:

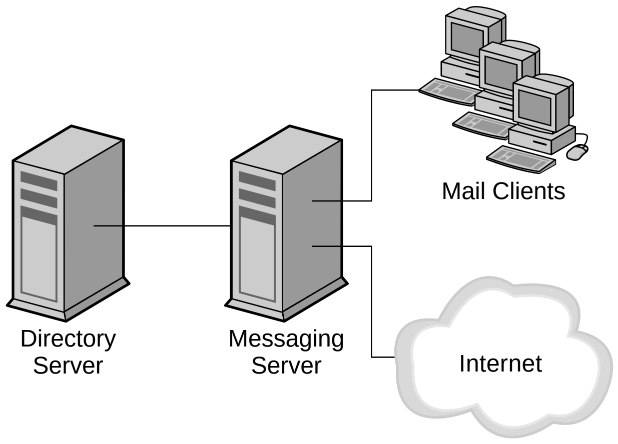
1. Add CPU or a hardware accelerator for SSL on MMP and MEM if appropriate.
2. Add memory to the machine if the MEM is being configured on it.
3. Add more disks for an SMTP proxy for the MMP.
4. Account for denial of service.
5. Add capacity for load balancing and redundancy, if appropriate.

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

Single-tiered Messaging Server Architecture

In a single-tiered architecture, there is no separation between access and data layers. The MTA, Message Store, and sometimes the Directory Server are installed in one layer. Figure 10-2 on page 168 shows a single-tiered architecture.

Figure 10-2 Simplified Messaging Server Single-tiered Architecture



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

➤ To Size a Single-tiered Messaging Server Architecture

1. Size your message stores like you size message stores in a "Two-tiered Messaging Server Architecture" on page 164.
2. Add CPU for SSL, if necessary.
3. Account for denial of service attacks.
4. Add more disks for the increased number of SMTP connections.
5. Add more disks for outbound MTA routing.

NOTE For specific instructions on sizing Messaging components in single-tiered or two-tiered architectures, contact your Sun Client Services representative.

Chapter 11

Developing a Messaging Server Architecture

This chapter describes how to design the architecture of your Messaging Server, which provides for how Messaging Server components are distributed across hardware and software resources.

This chapter contains the following sections:

- Understanding the Two-tiered Messaging Architecture
- Understanding Horizontal and Vertical Scalability in Messaging Server
- Planning for a Highly Available Messaging Server Deployment
- Performance Considerations for a Messaging Server Architecture

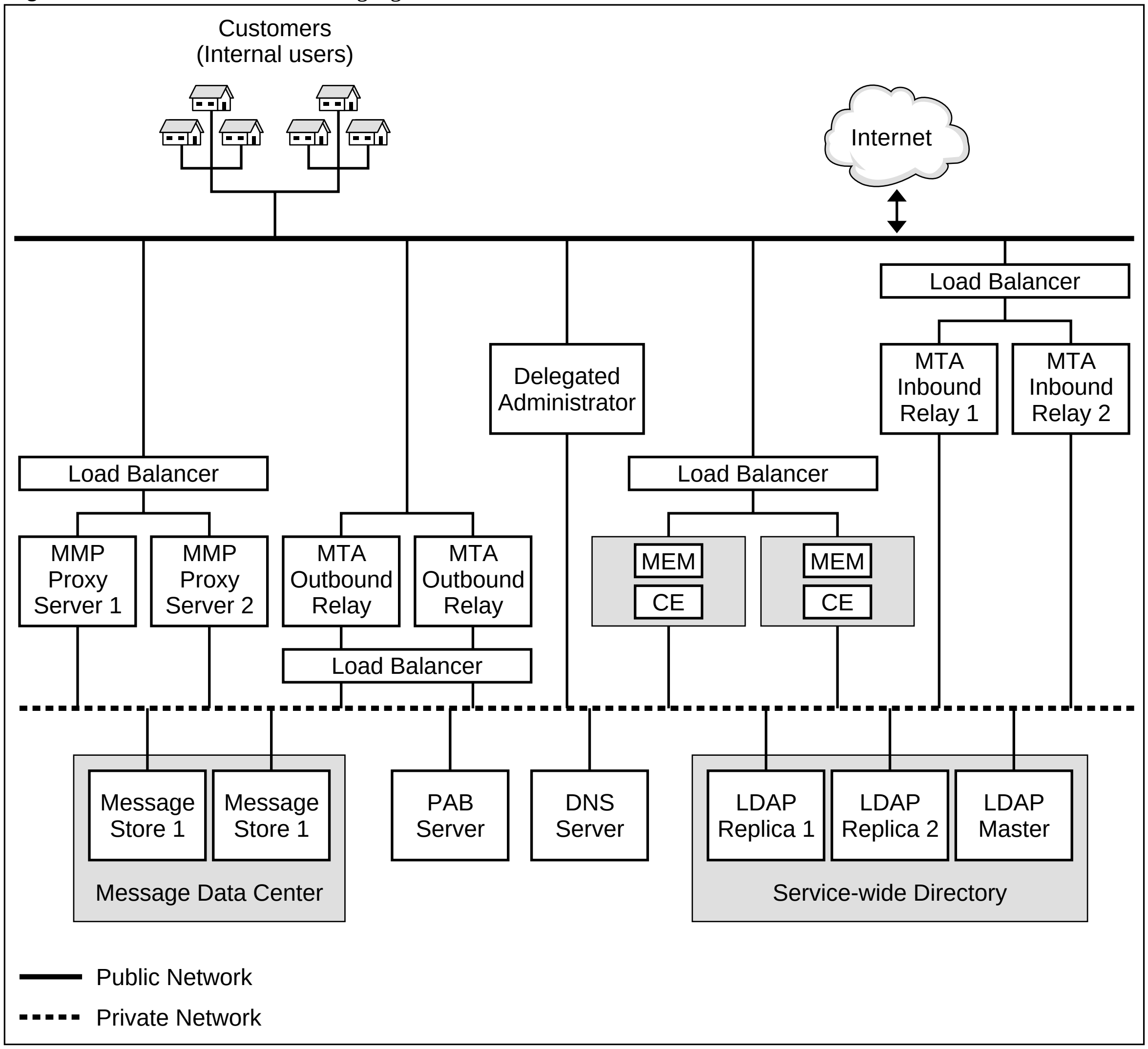
Understanding the Two-tiered Messaging Architecture

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

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

The following figure shows an example two-tiered architecture.

Figure 11-1 Two-Tiered Messaging Server Architecture



The following describes each of these functional pieces.

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

Private Data Network. This network provides secure connectivity between the public access network and Messaging Server data. It consists of a secure access layer and a data layer, which includes the service-wide directory, the message data center, and the 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 11-1 on page 170 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 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. Sometimes 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 Webmail (Messenger Express) servers. Figure 11-1 on page 170 shows a deployment that has two message stores. You can add more stores as needed.

Personal Address Book (PAB) Server. Stores and retrieves users’ addresses in an LDAP server, which can be the same server or a different server from the LDAP server described above.

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

Load Balancer. Balances network connections uniformly or by algorithm across multiple servers. Using load balancers, a single network address can represent a large number of servers, eliminating traffic bottlenecks, allowing management of traffic flows and guaranteeing high service levels. Figure 11-1 on page 170 shows load balancers for the MMPs, the MTAs, and the MEMs. Load balancers are not part of the Java Enterprise System product. You cannot use load balancers on the Message Store or directory masters. You use them for connections to MMPs, MEMs, Communications Express, MTAs, directory consumers, and with Messaging Server's MTA's use of the Brightmail product.

MTA Inbound Relay. MTA dedicated to accepting messages from external (Internet) sites and routing those messages to internal hosts and the local Message Store server. Because this is the first point of contact from the outside, the MTA inbound relay has the added responsibility of guarding against unauthorized relaying, spam filtering, and denial of service attack. You can use MX records to balance incoming mail traffic. See "Mail Exchange (MX) Records" on page 177 for more information.

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

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

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

Messenger Express Multiplexor or *MEM*. 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 Multiplexing Proxy (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. When you use multiple MEMs, they should be used with a load balancer. For Communications Express deployments, Communications Express software is also deployed on the same host that contains the MEM.

Two-tiered Architecture—Messaging Data Flow

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

Sending Mail: Internal User to Another Internal User

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

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

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

Retrieving Mail: Internal User

Synopsis: Internal User -> Load Balancer -> MMP / 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, or MEM/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 proxies for the rest of the connection between the client and servers.

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

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

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

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

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

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

Understanding Horizontal and Vertical Scalability in Messaging Server

Scalability is the capacity of your deployment to accommodate growth in the use of messaging services. Scalability determines how well your system can absorb rapid 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 overload resources of your existing deployment. Careful planning helps you to determine how to appropriately scale your deployment.

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

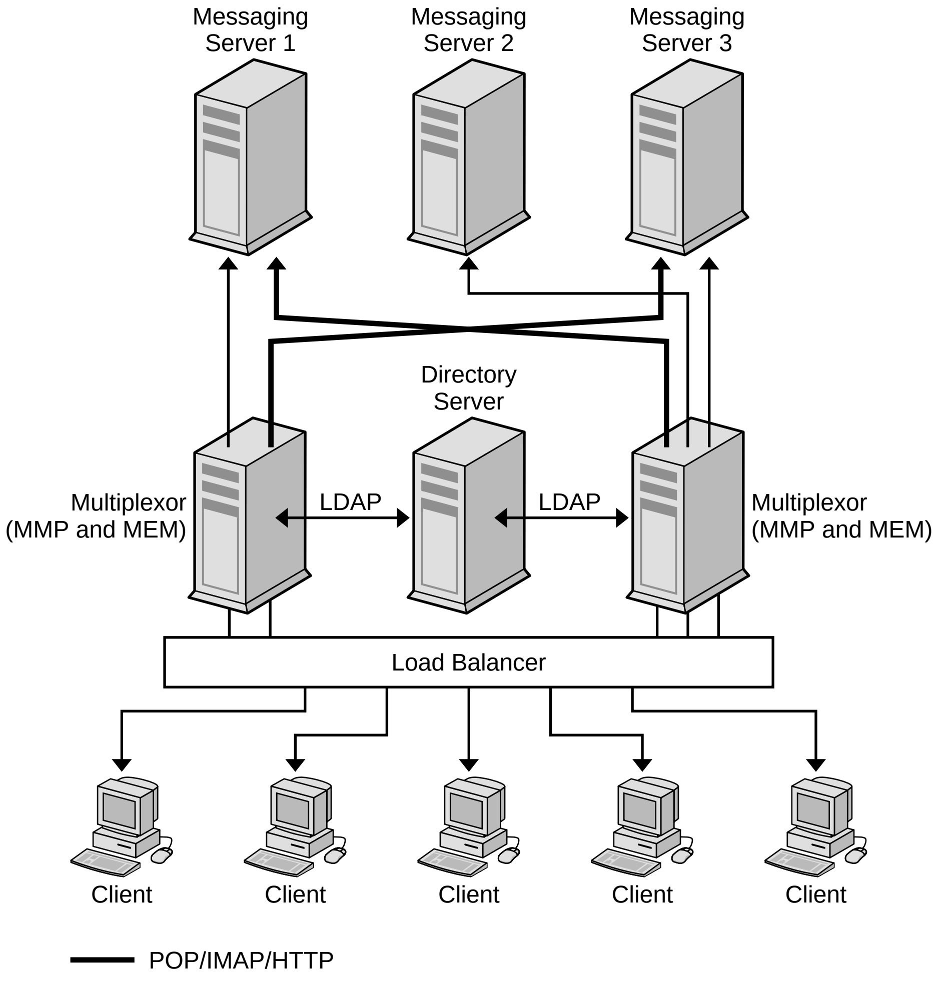
- Spreading Your Messaging User Base Across Several Servers
- Spreading Your Messaging Resources Across Redundant Components

Spreading Your Messaging User Base Across Several Servers

To distribute load across servers is to divide clients' mail evenly across several back-end Message Stores. You can divide up users alphabetically, by their Class of Service, by their department, or by their physical location and assign them to a specific back-end Message Store host.

Often, both the MMP and the MEM are placed on the same machine for ease of manageability. Figure 11-2 on page 176 shows a sample deployment where users are spread across multiple back-end servers and multiplexors enabled to handle incoming client connections.

Figure 11-2 Spreading Your User Base Across Multiple Servers



Spreading users across back-end servers provides simplified user management, as long as you use MMPs or MEMs. 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 Messaging Resources Across Redundant Components

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

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

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

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

Mail Exchange (MX) Records

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

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

Inbound and Outbound MTAs

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

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

Load Balancers

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

Planning for Vertical Scalability

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

To vertically scale your deployment, you need to:

- Size each messaging component

  See "Developing Messaging Server Architectural Strategies" on page 164.

- Test the load of a prototype of your system

  See "Using a Messaging Server Load Simulator" on page 161.

- Monitor system performance and adjust the deployment accordingly

Planning for a Highly Available Messaging Server Deployment

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 6, "Designing 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 topics:

- Message Store Performance Considerations
- MTA Performance Considerations
- MMP Performance Considerations
- MEM Performance Considerations
- Messaging Server and Directory Server Performance Consideration

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
8. Use of SSL

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

Messaging Server Directories

Messaging Server uses six 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 11-1 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: `/var/opt/SUNWmsgrsr/queue`

Table 11-1 High Access Messaging Server Directories *(Continued)*

High I/O Directory	Description and Defining Parameter
Messaging Server log directory	This directory contains log files which are constantly being appended with new logging information. The number of changes will depend on the logging level set. The directory location is controlled by the `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 the `IMTA_LOG` option in the `imta_tailor` file. Default location: `/var/opt/SUNWmsgsr/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 `/var/opt/SUNWmsgsr/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: `/var/opt/SUNWmsgsr/store/partition/primary`
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.`*partition_name*`.messagepath`, where *partition_name* 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.`*partition_name*`.messagepath`, where *partition_name* is the name of the partition. Default location: `/var/opt/SUNWmsgsr/store/partition/primary`
Mailbox list database temporary directory	The directory used by the Message Store for all temporary files. To maximize performance, this directory should be located under the fastest file system. For Solaris, use the `configutil` command to configure the `store.dbtmpdir` variable to a directory under `tmpfs`, for example, `/tmp/mboxlist`. Default location: `/var/opt/SUNWmsgsr/store/mboxlist`

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

MTA Queue Directories

In non-LMTP environments, the MTA queue directories in the Message Store system 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 two-tiered environment using LMTP, this directory will be lightly used, if at all. In prior releases of Messaging Server, on large systems this directory set needs to be on its own stripe or volume.

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

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

Multiple Store Partitions

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

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

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

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

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

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

Message Store Processor Scalability

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

Setting the Mailbox Database Cache Size

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`.

You should use the `configutil` parameter `store.dbtmpdir` to redefine the location of the mailbox database to `/tmp`, that is, `/tmp/mboxlist`.

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

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

Cache efficiency is measured by *hit rate*. Hit rate is the percentage of times that a database call can be handled by cache. An optimally sized cache will have a 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 so 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 database command `db_stat` can be used to measure the cache hit rate. In the following example, the `configutil` parameter `store.dbtmpdir` has redefined the location of the mailbox database to `/tmp`, that is, `/tmp/mboxlist`. The `db_stat` command is run against this location:

```
# /opt/SUNWmsgsr/lib/db_stat -m -h /tmp/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. 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 the `store.dbcachesize` parameter. The maximum size is the total of all the `*.db` files in the `store/mboxlist` directory. The cache size should not exceed the total size of all of the `.db` files under the `store/mboxlist` directory.

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 database.

Setting Disk Stripe Width

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

MTA Performance Considerations

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

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

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

MTA and RAID Trade-offs

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

MTA and Processor Scalability

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

MTA and High Availability

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

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

MMP Performance Considerations

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

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 container, 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.

MEM Performance Considerations

The MEM provides a middle-tier proxy for the Webmail (Messenger Express) 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.

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.

Messaging Server and Directory Server Performance Consideration

For large-scale installations with Access Manager, Messaging Server, and an LDAP Schema 2 directory, you might want to consolidate the Access Control Instructions (ACIs) in your directory.

When you install Access Manager with Messaging Server, a large number of ACIs initially are installed in the directory. Many default ACIs are not needed or used by Messaging Server. You can improve the performance of Directory Server and, consequently, of Messaging Server look-ups, by consolidating and reducing the number of default ACIs in the directory.

For information about how to consolidate and discard unused ACIs, see “Appendix D: ACI Consolidation” in the *Sun Java System Communications Services Delegated Administrator Guide*:

http://docs.sun.com/doc/819-0114

Chapter 12

Designing a Messaging Server Topology

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
- Designing a Messaging Topology
- Understanding Messaging Topology Elements
- Creating a Messaging Topology Example

Identifying Your Geographic Needs

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

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

- Mail transaction rate and volume of mail storage at each physical location

Designing a Messaging Topology

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

- Central Topology

  Consolidates most or all major system components and messaging servers at a single location.

- Distributed Topology

  Spreads most or all system components and messaging servers across multiple sites.

- Hybrid Topology

  Consolidates some system components and distributes other components across multiple locations.

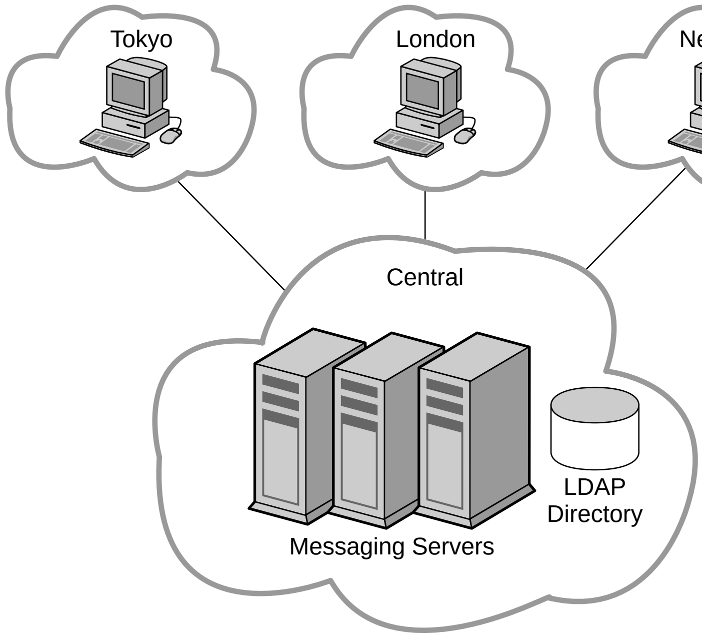
- Service Provider Topology

  Hosts multiple domains and handles larger customer base. Like a central topology, it consolidates most system components at a single location.

Central Topology

In a central topology, most or all major system components and messaging processes are located at one site. Clients at remote sites communicate over a Wide Area Network (WAN) to the centralized messaging servers. Figure 12-1 on page 191 shows a central topology.

Figure 12-1 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).

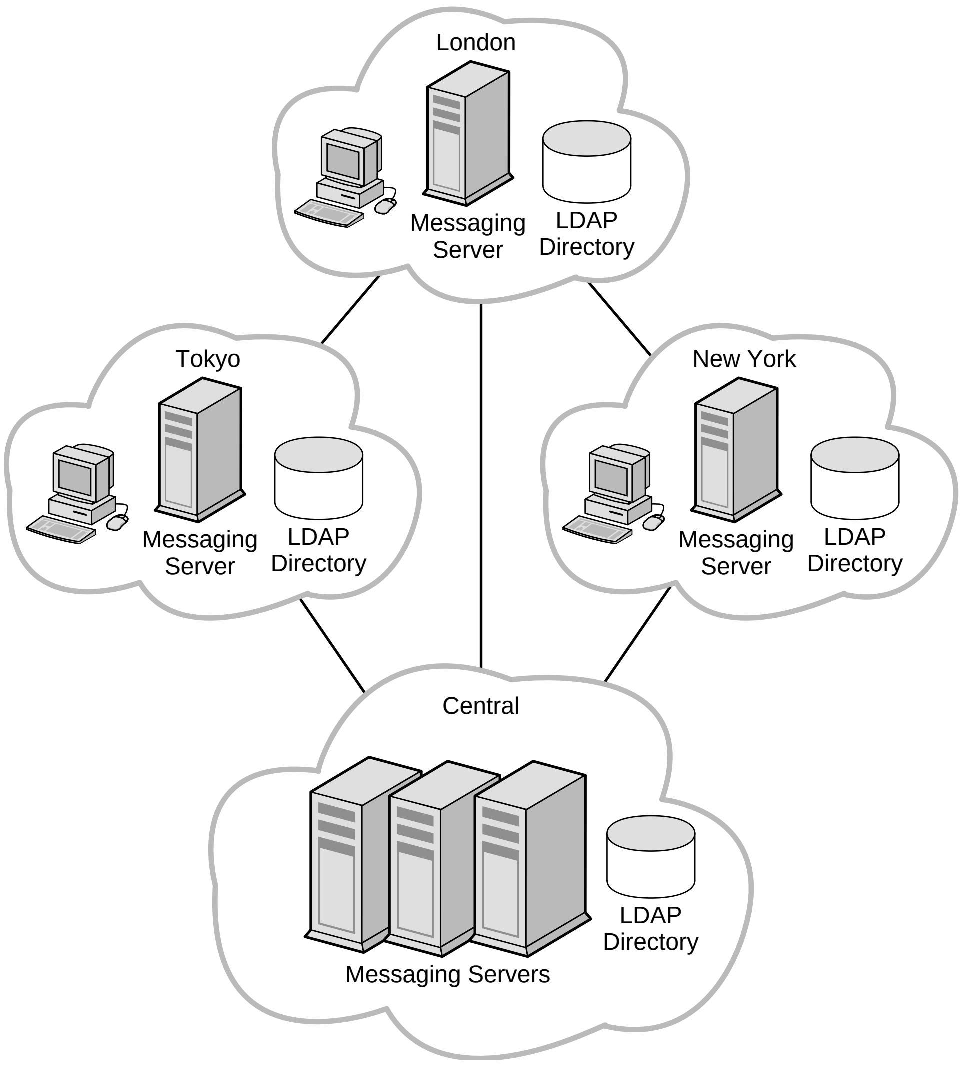
There are advantages to implementing a central topology. In general, a central topology has lower hardware and support costs. Central topologies tend to be easier to manage because you have a simplified messaging architecture and a directory replication structure with fewer replication agreements. With 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 12-1 on page 191, a message sent from one user in the Tokyo site would first travel to the Central site before being sent to another user in the Tokyo site.

Distributed Topology

In a distributed topology, most or all system components and messaging processes are distributed across multiple sites, usually at each remote site. Figure 12-2 on page 193 shows a distributed topology.

Figure 12-2 Distributed Topology



You should consider a distributed topology for your site when:

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

- There is poor bandwidth to remote sites.

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

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

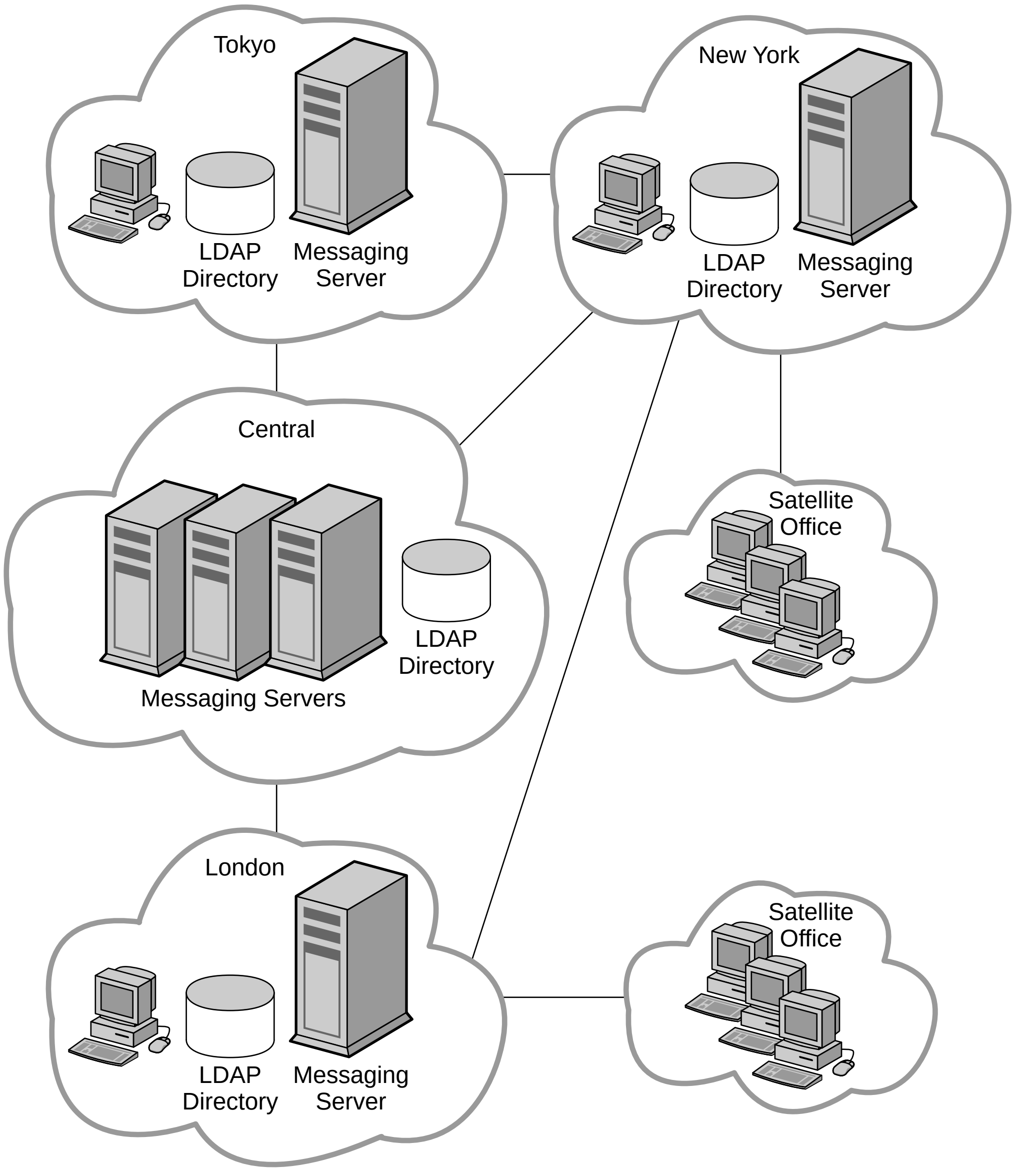
The disadvantages of implementing a distributed topology are that typically you will have higher hardware costs and higher support costs as you maintain more hardware at more locations. Support costs are also higher because of the complexity of the distributed topology. For example, failover in a distributed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

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

Hybrid Topology

In a hybrid topology, central and distributed topologies are combined to meet the needs of an organization. Figure 12-3 on page 195 shows a hybrid topology.

Figure 12-3 Hybrid Topology

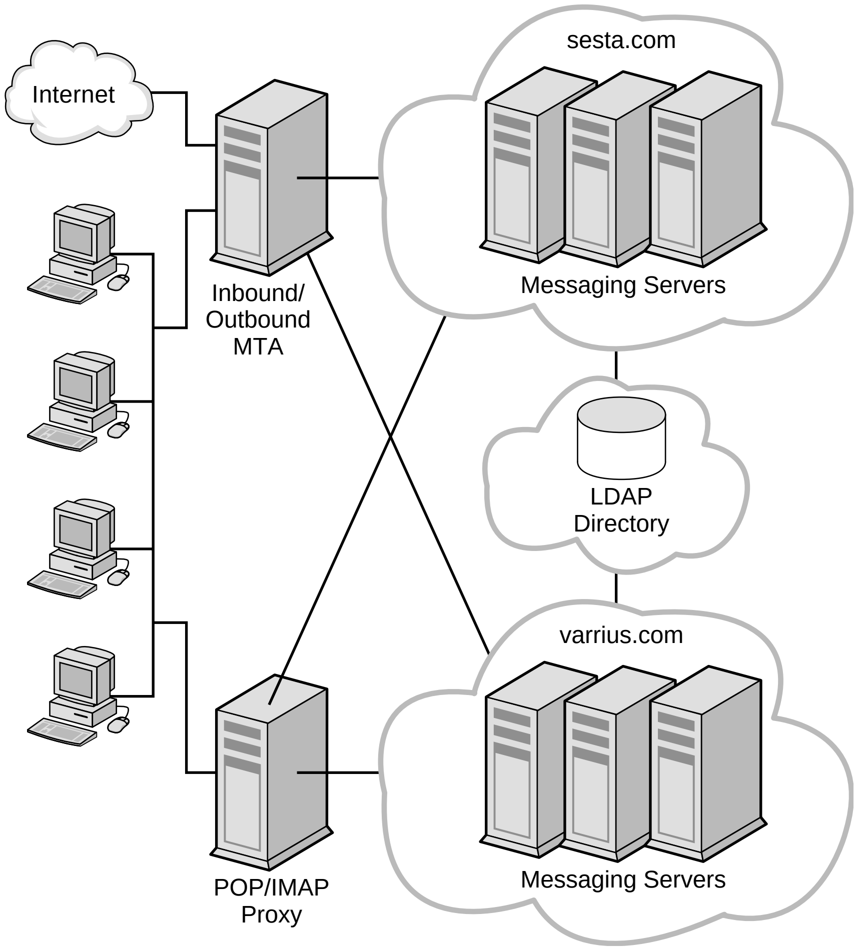


Organizations that benefit from a hybrid topology include those with many sites that have the ability to support a large user base. These sites that support them can house their own messaging servers. Some of these larger sites might have smaller satellite offices located in the general vicinity. But 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 12-4 on page 197 shows a service provider topology.

Figure 12-4 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
- Using MTAs to Protect Your Messaging System

- Using MMPs and MEMs
- Using Gateways

Messaging Topology Components

In "Designing a Messaging Topology" on page 190, you were introduced to three components of a messaging topology: Messaging Server, Directory Server, and clients. This section will describe other components in a basic messaging topology.

Messaging Server. Houses and maintains user mailboxes; it can also be a server that contains just the MTA portion of Messaging Server as described in **Internet-facing MTA** and **MTA Relay**.

Client. Accesses messaging services from Messaging Server (often through the Messaging Multiplexor).

Directory Server. Used by Messaging Server for name and alias lookup. Direct LDAP lookup determines where messages should be routed.

Messaging Multiplexor. Connects clients to the appropriate Messaging Server for retrieving messages.

Internet-facing MTA. Routes messages from the Internet and relays them across the firewall. Typically, a Messaging Server host is set up to perform this function.

MTA Relay. The inbound MTA routes incoming messages to valid addresses in the appropriate Messaging Server. The outgoing MTA accepts outgoing messages from clients, queries LDAP to find out where to send the message, then sends it off to the appropriate server or out across the firewall to the Internet. Typically, a Messaging Server host is set up to perform this function.

DNS Server. Resolves server names into IP addresses to allow messages to be routed to their proper address in the network.

Firewall. Restricts Internet access of your internal site. You might even have a firewall between departments in your organization.

Using MTAs to Protect Your Messaging System

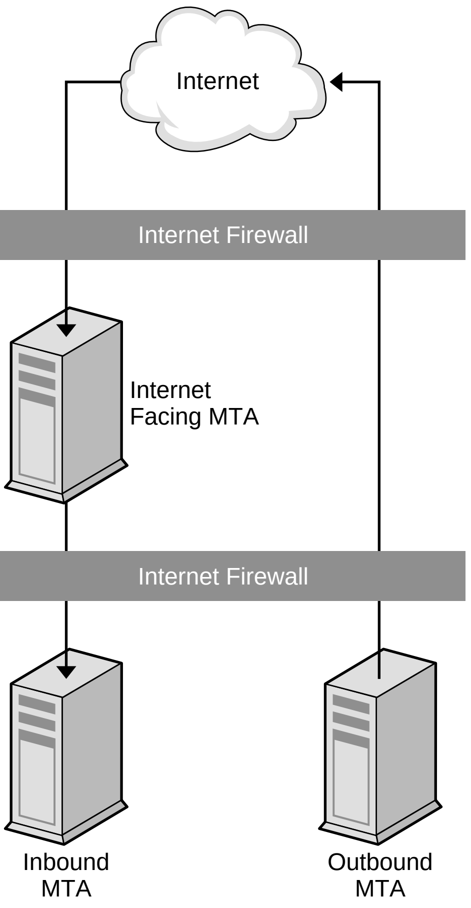
You can use MTAs to protect your Messaging Server deployment, as well as to control the flow of message traffic to and from your site.

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

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

The outbound MTA accepts outgoing messages from clients. It queries LDAP to find out where to send the message, then sends it off to the appropriate server or out across the firewall to the Internet. This offloads the MTA work from messaging servers that are used by users to retrieve messages. Figure 12-5 on page 200 illustrates the idea.

Figure 12-5 MTAs in Messaging Topology

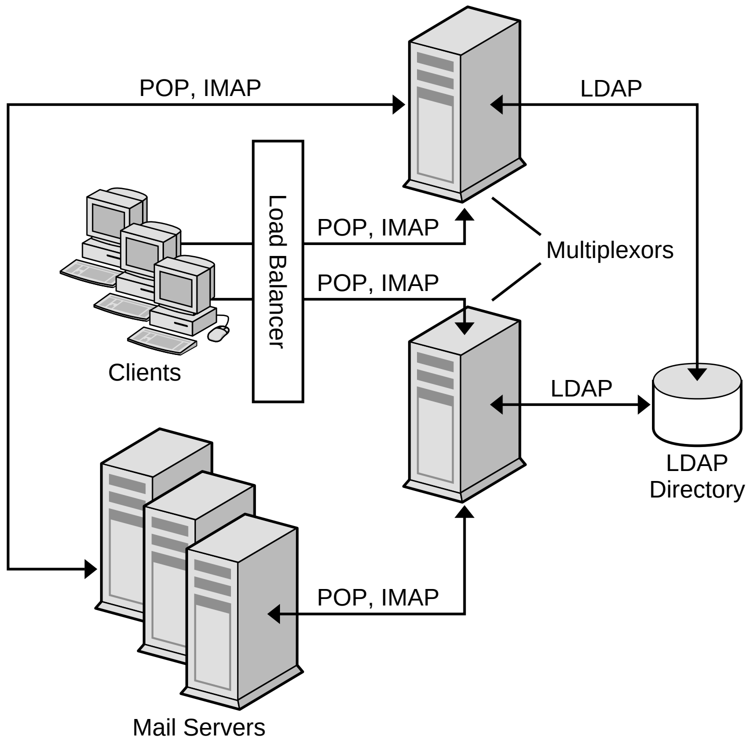


Using MMPs and MEMs

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

When such a client connects and authenticates, the MMP looks up the user information in the directory to determine where the user's messages are held. The MMP then connects the client to that specific server. 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 12-6 MMP Overview



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

Using Gateways

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

Creating a Messaging Topology Example

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

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

Step 1: Identifying Messaging Goals

The first step in creating a topology is to understand the goals of your organization. Similar to Chapter 2, "Analyzing Your Communications Services 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 Message Server hosts would be set up in Chicago, Los Angeles, and in New York. Since the volume of email traffic in San Diego and in Minneapolis is relatively light, these satellite offices will only have mail clients connecting to servers that are located in Chicago and in Los Angeles.

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 to the Internet.

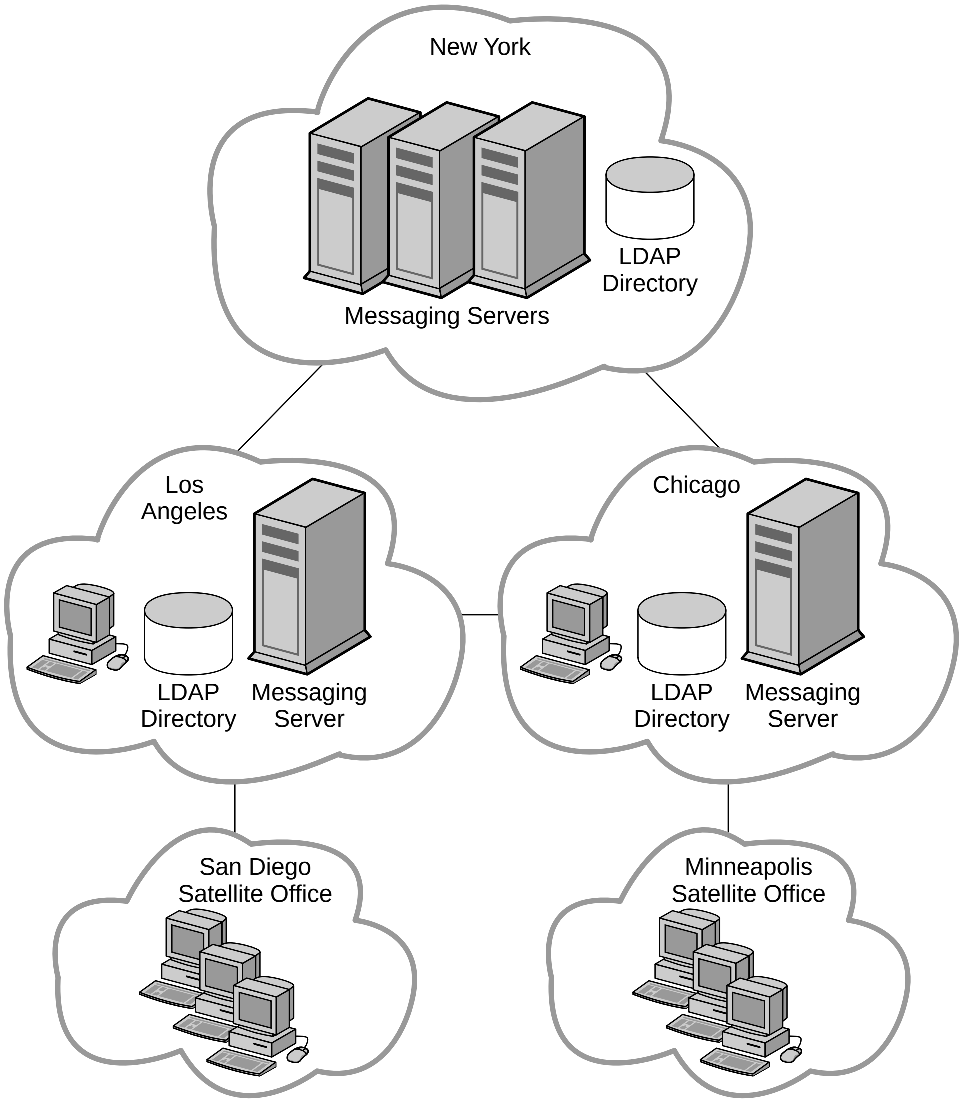
Step 2: Choosing a Topology Strategy

The second step in creating your messaging topology is to choose your topology strategy, described in “Designing a Messaging Topology” on page 190. The Siroe Corporation evaluated their business objectives as well as their financial and technical constraints. They determined that:

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

The Siroe Corporation then mapped their objectives and constraints to a common design strategy. Figure 12-7 on page 204 shows that the Siroe Corporation has chosen a hybrid topology.

Figure 12-7 Hybrid Topology for the Siroe Corporation

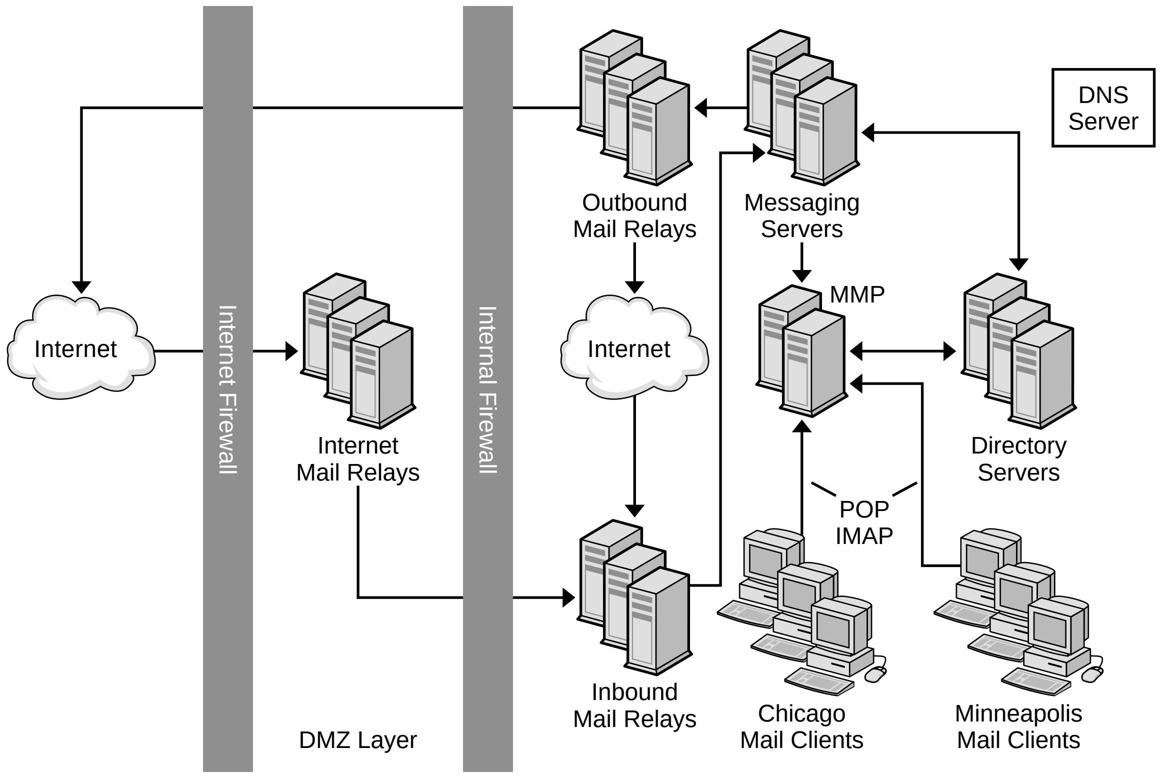


Because New York has the highest message transaction rate of messages entering and leaving the system, it has the most number of messaging servers. The smaller offices, Los Angeles and Chicago, also support San Diego and Minneapolis. However, these satellite offices do not require their own messaging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

Step 3: Planning the Topology Elements

The final step in creating your messaging topology is to plan your topology elements in your actual deployment, as described in "Understanding Messaging Topology Elements" on page 197. The following figure illustrates the topology elements in the Chicago and Minneapolis offices.

Figure 12-8 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 MTAs are placed in the topology to route messages from the Internet and relay them across the firewall. MTAs are added to route incoming and outgoing messages. Separating incoming and outgoing messages helps to manage the high volume of message traffic. The MMP connects employees' POP and IMAP mail clients to their mailboxes in the Messaging Servers. By using an MMP, employees 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 13

Planning Messaging Server Security

This chapter describes how to plan for and protect the various component of your Messaging Server deployment.

This chapter contains the following sections:

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 MTAs

Secure MTAs 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 MTAs.

Table 13-1 Common MTA 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*. Many components of the MTA employ table lookup-oriented information. This type of table is used to transform, that is, *map*, an input string into an output string. Mapping tables are usually presented as two columns. The first (left-hand) column provides possible input strings against which to match (pattern), and the second (right-hand) column gives the resulting output string for which the input string is mapped (template).

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:

`http://docs.sun.com/doc/819-0105`

Table 13-2 Access Control Mapping Tables

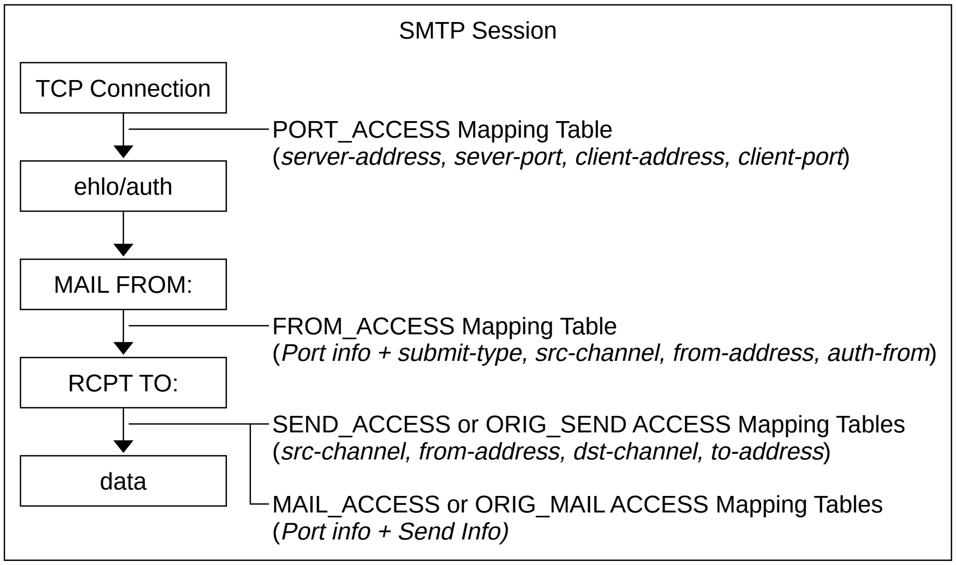
Mapping Table	Description
`SEND_ACCESS`	Used to block incoming connections based on envelope `From:` address, envelope `To:` address, source and destination channels. The `To:` address is checked after rewriting, alias expansion, and so on, have been performed.
`ORIG_SEND_ACCESS`	Used to block incoming 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`.

Table 13-2 Access Control Mapping Tables *(Continued)*

Mapping Table	Description
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 13-1 Mapping Tables and the Mail Acceptance Process


SMTP Session
TCP Connection
PORT_ACCESS Mapping Table
(*server-address, sever-port, client-address, client-port*)
ehlo/auth
MAIL FROM:
FROM_ACCESS Mapping Table
(*Port info + submit-type, src-channel, from-address, auth-from*)
RCPT TO:
SEND_ACCESS or ORIG_SEND ACCESS Mapping Tables
(*src-channel, from-address, dst-channel, to-address*)
data
MAIL_ACCESS or ORIG_MAIL ACCESS Mapping Tables
(*Port info + Send Info)*

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 clients do not authenticate by using SMTP AUTH, as described in "Enabling Authenticated SMTP" on page 220, 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*.

   `http://docs.sun.com/doc/819-0105`

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:

   `http://docs.sun.com/doc/819-0105`

   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:

   `http://docs.sun.com/doc/819-0105`

   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*:

http://docs.sun.com/doc/819-0107

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*:

http://docs.sun.com/doc/819-0105

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 MTAs 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*.

`http://docs.sun.com/doc/819-0105`

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 Messaging User Authentication" on page 218.

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-tiered architecture is recommended over a one-tiered 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-tiered 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 MTAs handle that processing. In addition, user access to the store is limited to through an MMP or an MEM in a two-tiered deployment, potentially adding an extra security layer to the message store.

If you deploy a one-tiered 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 10, "Planning a Messaging Server 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*.

`http://docs.sun.com/doc/819-0105`

Protecting MMPs and MEMs

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 220.

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 MTAs, 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 Messaging 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*:

`http://docs.sun.com/doc/817-7607`

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*:

http://docs.sun.com/doc/819-0106

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 13-3 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	-	-
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 222). The SASL mechanisms, PLAIN 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 220) 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*:

`http://docs.sun.com/doc/819-0105`

Enabling Authenticated SMTP

By default, users need not submit a password when they connect to the SMTP service of Messaging Server to send a message. You can, however, enable password login to SMTP in order to enable 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.

If you require SMTP AUTH for mail submission, 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*:

`http://docs.sun.com/doc/819-0105`

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 222.

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.

The following table describes the SSL authentication support with each client access protocol.

Table 13-4 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

The SMTP, POP, and IMAP 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, and IMAP servers can also be configured to use SSL on an alternate port, for clients which 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 also have any number of certificates of trusted Certificate 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*:

`http://docs.sun.com/doc/819-0105`

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 requires the use of S/MIME. See "Signed and Encrypted S/MIME" on page 224 for more information.

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 encrypted by a stronger cipher is more difficult for an unauthorized person to decrypt.

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 know 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*:

`http://docs.sun.com/doc/819-0105`

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. The recipients can store the encrypted messages after receipt, decrypting them only to read them.

Communications Express Mail now includes the security advantages of S/MIME. Communications Express Mail users who are set up to use S/MIME can exchange signed or encrypted messages with other Communications Express Mail users, and with users of the Microsoft Outlook mail system or other mail clients that support S/MIME. See “Requirements for Using S/MIME with Communications Express Mail” on page 350 for more information.

For other clients that support S/MIME, see that client documentation for information on S/MIME configuration.

Chapter 14

Planning a Messaging Server 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.** Uses 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. For more information, see the *Sun Java System Messaging Server Adminstration Guide*:

`http://docs.sun.com/doc/819-0105`

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 209 for more information.

Mailbox Filtering

Messaging Server provides 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 212 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 230 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.

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 211 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.

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 decide to use 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 mail. 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/819-0105`

Using Symantec 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 might 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 connecting to the Brightmail spam and virus scanning library can be used to connect to the SpamAssassin `spamd` server.

Using Symantec AntiVirus Scan Engine (SAVSE)

Messaging Server supports the use of SAVSE. SAVSE is a TCP/IP server application and communications API that provides high-performance virus scanning. It is designed to protect traffic served through, or stored on, network infrastructure devices.

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 15

Understanding Messaging Server 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 2005Q1 Installation Guide* for instructions on running the Java Enterprise System installer:

`http://docs.sun.com/doc/819-0056`

This chapter contains the following sections:

- Messaging Server Installation Considerations
- Messaging Server Installation Worksheets
- Choosing Which Messaging Server Components to Configure
- Disabling the sendmail Daemon

Messaging Server Installation Considerations

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

- **Resource Contention.** To avoid resource contention between 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 `root`.

- **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, Access Manager, 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 15-1 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	Access Manager (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:

http://docs.sun.com/doc/819-0106

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
```

Messaging Server 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*:

`http://docs.sun.com/doc/819-0105`

Table 15-2 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 15-2 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.	`Admin`	Administration Server Configuration	
	Password for the Administrator.	`PaSsWoRd`		
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 `root` in UNIX. Typically, this administrator is responsible for user and group data.	`cn=Directory Manager`	`comm_dssetup.pl` Perl script and Messaging Server Configuration	
	Password for the Directory Manager.	`pAsSwOrD`		
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 237 to answer certain questions.

Table 15-3 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.	`Admin`	
	Password of Administrator ID.	PaSsWoRd	

Table 15-3 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 15-4 Which Messaging Server Components to Configure?

Type of Messaging Host Being Configured	Needs These Components Selected in the Configurator Program
MTA	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 15-4 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 MTAs and on the back-end stores. See the *Sun Java System Messaging Server Administration Guide* for instructions on configuring LMTP:

`http://docs.sun.com/doc/819-0105`

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.

NOTE In some cases (especially on Solaris 10), even after you run the `/etc/init.d/sendmail stop` command, `sendmail` is autorestarted. In this case, use the following command to stop the `sendmail` process:

```
svcadmin disable network/smtp:sendmail
```

Part III

Deploying Calendar Server

Chapter 16

Introduction to Calendar Server Software

This chapter provides an overview of Sun Java System Calendar Server, the business reasoning behind deploying Calendar Server, and the deployment process itself.

This chapter contains the following sections:

- Calendar Server Overview
- Designing Your Calendar Server Deployment

Calendar Server Overview

Sun Java System Calendar Server (formerly Sun™ ONE Calendar Server) is a high-performance, Internet standards-based calendar server designed with the scalability to meet the needs of customers ranging from medium- and large-sized enterprises to even the largest telecommunication and Internet service providers. Through a native Web browser interface or connectors to other calendar clients, including Microsoft Outlook, Calendar Server provides group scheduling and personal calendaring to consumers at home or at work, while enabling them to share calendar information with others over the Internet. The user interface (UI) can be customized to include Web links for e-commerce, banner ads, logo, or brand of the calendar server customer, and more.

Calendar Server provides one of the industry’s most open, interoperable, and high-performance time and resource management solutions. Through its scalability, performance, and reliability, it provides the features you require at a lower total cost of ownership than alternative solutions. Native support for iCalendar standards enables users to schedule events in a format that is easily shared across the Internet. Calendar Server employs standards and protocols such as:

- Internet Calendaring (iCalendar)
- iCalendar Transport-Independent Interoperability Protocol (iTIP)
- iCalendar Message-based Interoperability Protocol (iMIP)
- eXtensible Markup Language (XML)
- Lightweight Directory Access Protocol (LDAP)
- HyperText Transport Protocol (HTTP)

The Calendar Server architecture is flexible, extensible, and scalable both vertically (by increasing the number of CPUs per system) and horizontally (by introducing additional servers into the network). As a result, Calendar Server can be thought of as a system of servers that can be configured to fit a variety of needs. It can remain in isolation as a standalone calendar server, or it can be configured with many instances, having the various services duplicated or split between them.

Calendar Server makes use of plugins to obtain external services. Calendar Server also supports both LDAP- and identity-based deployments, and integrates with Sun Java System Access Manager (formerly Identity Server), Sun Java System Portal Server (formerly Sun ONE Portal Server), and Sun Java System Instant Messaging (formerly Sun ONE Instant Messaging) to provide additional functionality.

Calendar Server provides the following benefits:

- Web-based group scheduling, free-busy lookup, and corporate directory lookup
- Web-based resource scheduling for conference rooms, projectors, and other resources
- XML-based customization (color, login, user interface, logo, branding, and so on)
- Support for standards-based event and calendar data feeds published in XML or iCalendar formats, which improves communication and can offer new revenue opportunities through commerce links and banner ads

- Native LDAP support, with an API for other directory services
- Connectors to additional calendar clients, such as Microsoft Outlook, enabling these clients to perform scheduling on Calendar Server
- Support for hosted domains
- Simplified system management, online backup and restore, and entire database backup and restore
- Support for multiple calendars, such as work, family, friends, and more
- Support for public and private calendars, as well as public, private, and confidential individual events
- Support for layered calendar views, which enables users to combine two or more calendars into a single view, providing improved communication and productivity
- Automatic e-mail notification of appointments, invitations, and reminders sent to selected recipients; integrates with Sun Java System Instant Messaging to provide automatic pop-up reminders
- Support for multiple owners of each calendar, to facilitate communication and productivity with project teams and community groups
- Ability to delegate calendar ownership to others who may act on behalf of the primary owner
- Daily, weekly, monthly, yearly, and comparison views
- Supports hundreds of thousands of users through a scalable, networked, server-to-server, client- server architecture
- Supports Secure Sockets Layer (SSL) encryption, LDAP authentication, authentication plugins, and identity-enabled single sign-on (SSO) through Access Manager

For more information on Calendar Server concepts, see the *Sun Java System Calendar Server Administration Guide*:

`http://docs.sun.com/doc/819-0024`

Designing Your Calendar Server Deployment

The deployment process consists of the following general phases, referred to as the Solution Life Cycle:

- Analyzing business requirements
- Analyzing technical requirements
- Designing the logical architecture
- Designing the deployment architecture
- Implementing the deployment
- Operating the deployment

The deployment phases are not rigid: the deployment process is iterative in nature.

For detailed information on the deployment process for Calendar Server, and other Java Enterprise System components, see the *Sun Java System Deployment Planning Guide*:

http://docs.sun.com/doc/819-0058

Objectives of Your Calendar Server Deployment

Before you begin your Calendar Server deployment planning, a good question to ask is:

Why is my organization deploying Calendar Server?

Several reasons to consider are:

- **Cost savings.** The total cost of ownership per user is lower than using other calendar products on the market.
- **Increased productivity.** Your calendar users can manage their events and tasks as well as schedule meetings and appointments with others in the organization. Your users can also manage calendar groups and resources such as meeting rooms and equipment. They can also synchronize their calendars with mobile devices (PDAs) and Microsoft Outlook.
- **Improved scalability and availability.** Calendar Server scales both horizontally and vertically. If your organization grows, you can easily upgrade your configuration by upgrading a server or add more servers.

- **Improved security.** If you deploy Calendar Server on a Solaris system, your organization can avoid many viruses and other security threats that are common in Windows environments.
- **High availability (HA) configuration.** Integration with e Sun Cluster software enables you to configure Calendar Server as a highly available service. If you experience a software or hardware failure, Calendar Server fails over to a secondary server.

Calendar Server Deployment Team

Deploying Calendar Server usually involves a number of people, each with different roles and responsibilities. In a small organization, one person might perform several roles. Some of the roles to consider are:

- Program Manager oversees the overall Calendar Server deployment and is responsible for its success or failure.
- Calendar Server Administrator performs day-to-day administrative tasks to manage Calendar Server and might also be responsible for installing and upgrading Calendar Server.
- Performance Engineer tests and monitors the Calendar Server performance for the trial and production deployments to see if the deployment criteria is met.
- Development Engineering writes Calendar Server applications or plugins, or customizes the Calendar Server user interface (UI), if required.
- Documentation Specialist writes any customized documentation for administrators and end users.
- Education/Training develops training classes and material.
- Support Specialists, who support both the trial and production deployments.

Calendar Server End Users

End users can connect to Calendar Server by using the Calendar Express Web client, Communications Express web client, or Sun Java System Connector for Microsoft Outlook.

Questions about end users at your site include:

- How many total Calendar Server end users will your site have?

- How will your end users connect to Calendar Server? Calendar Express? Communications Express? Microsoft Outlook? A combination of clients?
- How many geographic locations are involved? Are your end users all in the same or different time zones?
- Will your end users log into Calendar Server at the same time each day?
- How many active end users will your deployment have during peak use?
- How fast will your end user base grow?
- What are your specific performance requirements for Calendar Server end users?
- What are your single sign-on (SSO) requirements?
- Are any of your users migrating from Netscape™ Calendar 4.x?
- Are your end users planning to use Sun ONE Synchronization?
- Are you planning to customize the Calendar Server UI for your end users?
- Does your site plan to use a proxy server?
- Does your site plan to use load balancing?

Expected Calendar Server End User Performance

What are your specific performance requirements for your end users? For example:

- What end user response times are acceptable?
- Can you tolerate a possible degradation in performance during peak load times?

What configuration do you plan to use for your deployment? Calendar Server configuration scenarios include:

- Single Calendar Server instance
- Single front-end server with single back-end database server
- Multiple front-end servers with multiple back-end database servers using LDAP CLD plugin
- Multiple front-end / back-end servers using LDAP CLD plugin

- High Availability (HA) configuration

If you plan to configure multiple front-end servers, how do you plan to distribute your end users?

If you plan to configure multiple back-end database servers, how do you plan to distribute your database? For example, you could distribute servers geographically.

What plans to you have for growth? For both front-end and back-end servers?

Chapter 17

Developing a Calendar Server Architecture

This chapter describes three basic Calendar Server deployment architectures, which can vary depending on your site's specific requirements.

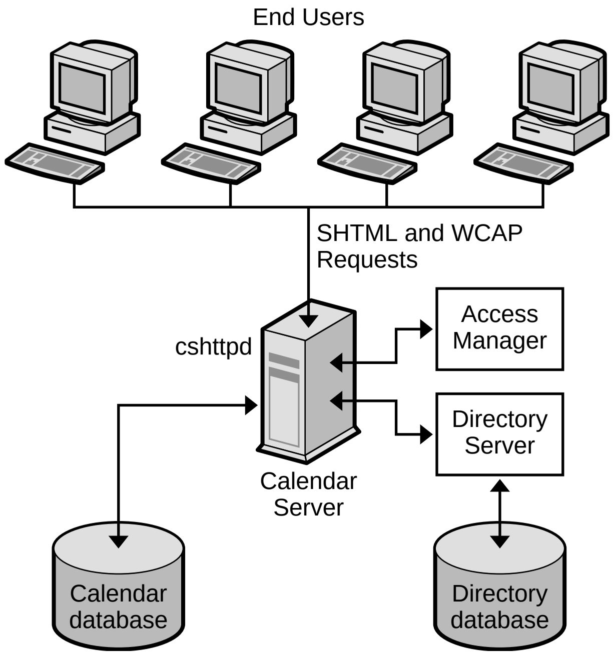
This chapter contains the following sections:

- Single-Server Calendar Server Architecture
- Two-tiered Calendar Server Architecture
- Two-tiered, Multiple Server Calendar Server Architecture

Single-Server Calendar Server Architecture

Figure 17-1 on page 254 shows a single-server architecture. In this deployment, all Calendar Server services (processes) run on the same server, either in the same CPU (processor) or across multiple CPUs. The Directory Server and Access Manager processes can run on the same server or on different servers.

Figure 17-1 Single-Server Calendar Server Architecture



A Calendar Server instance on a single server includes the following services:

- Administration service (`csadmind` process) provides support for administration functions such as commands to start or stop Calendar Server, create or delete calendar users or resources, or fetch and store calendars.
- HTTP service (`cshttpd` process) handles incoming SHTML and WCAP requests.
- Event Notification Service (`enpd`) acts as the broker for event and alarm notifications.
- Backup service (`csstored`) implements automatic backups, both archival backups and hot backups.

For a description of Calendar Server services, see the *Sun Java System Calendar Server Administration Guide*:

`http://docs.sun.com/819-0024`

The Database Wire Protocol (DWP) service (`csdwpd` process), which provides networking capability when the calendar database is on another server, is not required for a minimal configuration because the database is on the same server.

Calendar Server requires a directory server to authenticate users and to store user preferences. Usually, the directory server is an LDAP directory server, such as Sun Java System Directory Server. However, if you prefer, you can use the Calendar Server API (CSAPI) to write a plugin to use a non-LDAP directory server. This API is described in the *Sun Java System Calendar Server Developer's Guide*:

`http://docs.sun.com/doc/819-0025`

The directory server can run on the same server where Calendar Server is running or on a remote server.

Sun Java System Access Manager (release 2003Q4 (6.1) or later) provides the following functionality:

- **Communications Services Delegated Administrator utility.** Use the CLI utility (`commadmin`) to provision and manage hosted (virtual) domains, users, groups, organizations, resources, and roles for Sun Java System communications servers, including Calendar Server.

  For information about the Communications Services Delegated Administrator utility, see the *Sun Java System Communications Services Delegated Administrator Guide*:

  `http://docs.sun.com/doc/819-0114`

- **Single Sign-on (SSO).** You can implement SSO for Sun Java Enterprise System servers, including Calendar Server and Messaging Server, using Access Manager, or through trusted circle technology. Access Manager serves as the SSO gateway for Java Enterprise System servers. Users log in to Access Manager and then can access other the servers, as long as all servers are configured properly for SSO.

- **Sun Java System LDAP Schema 2.** Access Manager (release 2003Q4 or later) is required if you want to use this version of the schema.

For more information on these topics, see the *Sun Java System Calendar Server Administration Guide*:

`http://docs.sun.com/819-0024`

Access Manager can run on the same server where Calendar Server is running or on a remote server.

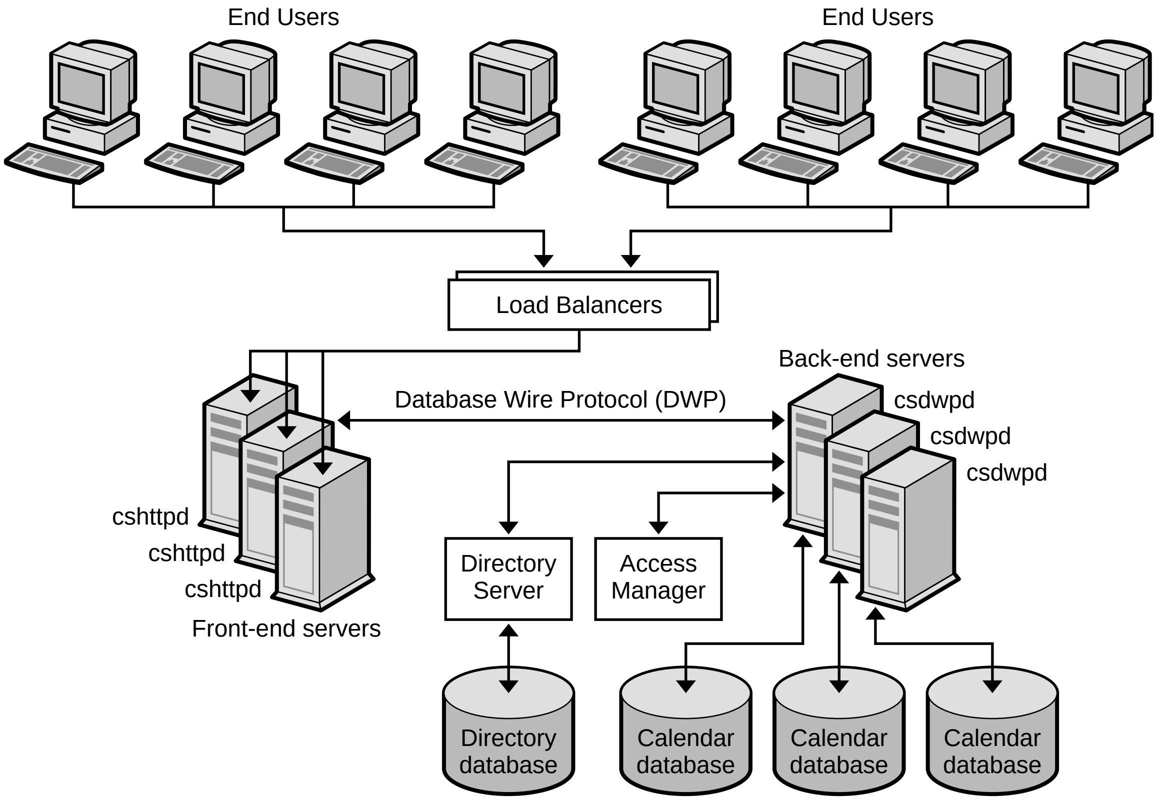
End users connect to Calendar Server from client machines by using one of two Web user interfaces (UIs), either Sun Java System Calendar Express, or Sun Java System Communications Express. For information using these interfaces, see the respective interface's online Help.

Two-tiered Calendar Server Architecture

Calendar Server supports scalability by distributing a configuration over multiple front-end and back-end servers. On each server, Calendar Server services can also be distributed across multiple CPUs.

In a two-tiered architecture, sometimes referred to as network front-end / database back-end configuration (shown in the following figure), users log in to a front-end server and connect to a back-end server using the Database Wire Protocol (DWP) service (`csdwpd` process). The calendar database is connected only to the back-end servers.

Figure 17-2 Two-tiered Calendar Server Architecture



Calendar Server processes run on both the front-end and back-end servers as follows:

- Users are directed by load balancers to a front-end server, where they log in. Each front-end server requires these services:
  - Administration Service (`csadmind` process)
  - HTTP Service (`cshttpd` process)
- Each back-end server is connected to a calendar database, so each back-end server requires these services:
  - Administration Service (`csadmind` process)
  - Backup service (`csstored`)
  - Event Notification Service (`enpd` and `csnotifyd` processes)
  - Database Wire Protocol (DWP) Service (`csdwpd` process) to provide networking capability to the front-end servers for the calendar database

In this configuration, users do not log in to the back-end servers, so the HTTP service (`cshttpd` process) is not required.

For a description of Calendar Server services, see the *Sun Java System Calendar Server Administration Guide*:

`http://docs.sun.com/819-0024`

A scalable Calendar Server configuration requires a directory server to authenticate users and to store user preferences.

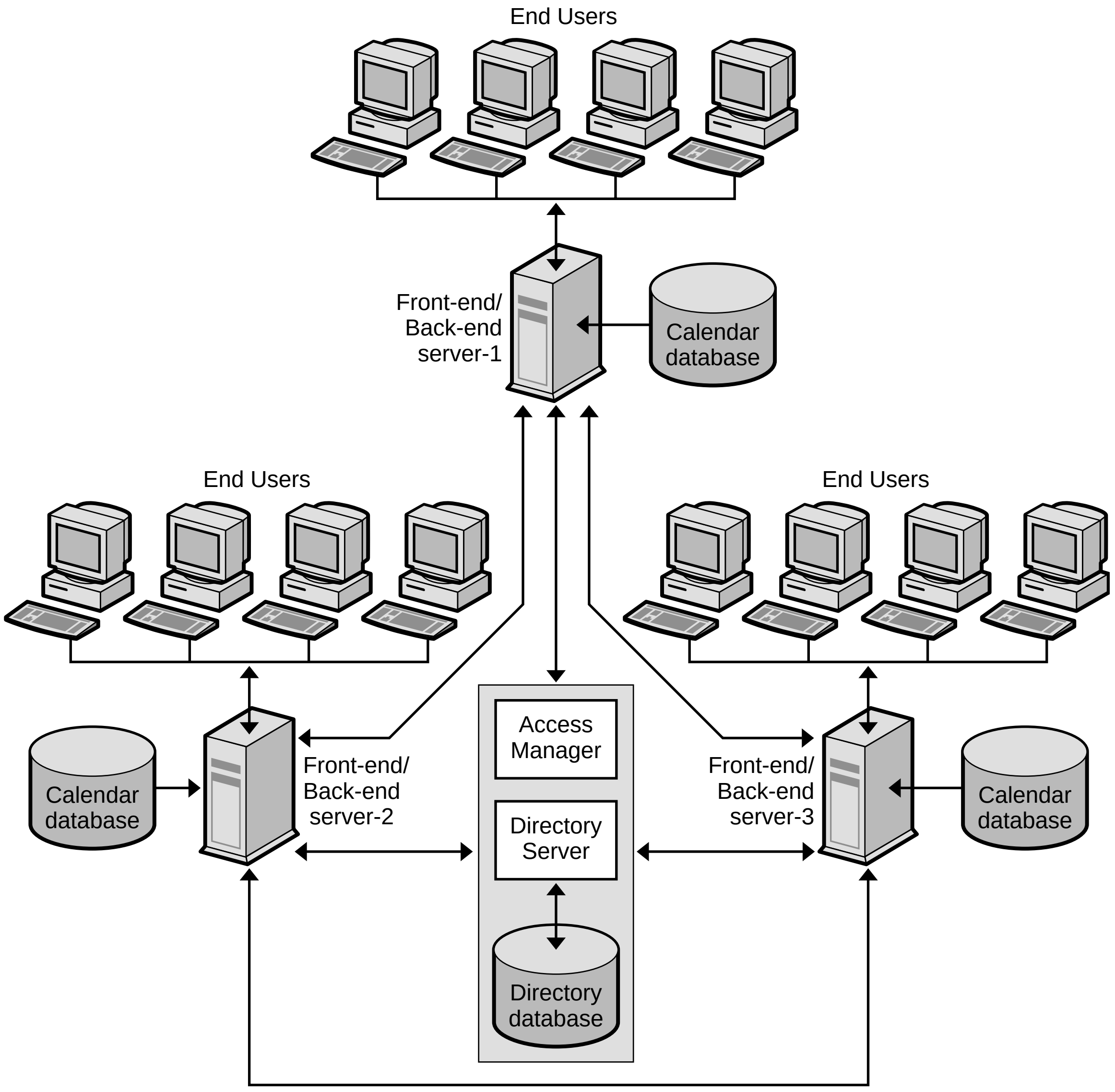
You can use Access Manager (release 6.1 (release 6 2003Q4) or later) to implement Single Sign-on (SSO), to use Sun Java Enterprise System LDAP Schema 2, or to provision and manage hosted (virtual) domains, users, groups, organizations, resources, and roles.

End users connect to Calendar Server from client machines by using one of two Web user interfaces (UIs), either Sun Java System Calendar Express, or Sun Java System Communications Express. For information using these interfaces, see the respective interface’s online Help.

Two-tiered, Multiple Server Calendar Server Architecture

In a two-tiered Calendar Server architecture with multiple front-end and back-end servers (shown in Figure 17-3 on page 259), users log in to a specific server, and each server is connected to a calendar database. This configuration enables calendars to be geographically distributed, with each calendar residing on the server where its owner logs in to Calendar Server.

Figure 17-3 Two-tiered, Multiple Server Calendar Server Architecture



In this architecture, each server functions as both a front end and back end, and requires all Calendar Server services: Administration Service (`csadmind` process), HTTP Service (`cshttpd` process), Event Notification Service (`enpd` and `csnotifyd` processes), and Database Wire Protocol (DWP) Service (`csdwpd` process).

For a description of Calendar Server services, see the *Sun Java System Calendar Server Administration Guide*:

http://docs.sun.com/doc/819-0024

NOTE In this architecture, you could also separate the front end services from the back end services onto separate machines, and use the LDAP Calendar Lookup Database (CLD) to determine which back end a front end needs to get data from. For more information, see the *Sun Java System Calendar Server Administration Guide*.

A multiple front-end / back-end server deployment requires a directory server to authenticate users and to store user preferences.

You can use Access Manager (release 6.1 (release 6 2003Q4) or later) to implement Single Sign-on (SSO), to use Sun Java Enterprise System LDAP Schema 2, or to provision and manage hosted (virtual) domains, users, groups, organizations, resources, and roles.

End users connect to Calendar Server from client machines by using one of two Web user interfaces (UIs), either Sun Java System Calendar Express, or Sun Java System Communications Express. For information using these interfaces, see the respective interface’s online Help.

Chapter 18

Planning Calendar Server Security

This chapter describes how to plan for and protect the various components of your Calendar Server deployment.

This chapter contains the following sections:

- Calendar Server Security Overview
- Planning Calendar User Authentication

Calendar Server Security Overview

Security plays a key role in the day-to-day operations of today's businesses. Breaches in security can not only compromise trade secrets, but can also result in downtime, data corruption, and increased operation costs. Calendar Server provides a number of security levels to protect users against eavesdropping, unsanctioned usage, or external attack. The basic level of security is through authentication. Calendar Server uses LDAP authentication by default, but also supports the use of an authentication plugin for cases where an alternate means of authentication is desired. Furthermore, integration with Access Manager enables Calendar Server to take advantage of its single sign-on capability.

Security involves not only ensuring the integrity of users. It also means ensuring the confidentiality of data. To this end, Calendar Server supports the use of SSL encryption for login, or both login and data. In other words, only the login may be encrypted, or the entire session including the login may be encrypted, from the Web client to the server.

Integration with Secure Remote Access also provides SSL encryption, but through a proxy gateway. In addition, integration with the portal gateway provides a URL rewriting capability to further insulate Calendar Server from external entities. Calendar Server can be deployed with the portal gateway such that there is no direct connection to the Calendar Server without going through the gateway. In this case, every URL is rewritten, thus obfuscating the true URL of the Calendar Server. Even though a user is authenticated, that does not mean that the user should have access to other calendar users' data.

Within a calendar domain exist other layers of security to prevent authenticated users from unauthorized access to other authenticated users' calendar data. One security measure is through the Calendar Server access control entries. Access control enables calendar users to specify who can see their calendars, who can schedule events into their calendars, who can modify their calendars, and who can delete events from their calendars. Access control also enables users to select who can act on their behalf to respond to invitations, schedule or modify events, and delete events. Finally, access control can be used to span domains of users, thus preventing (or enabling) users in one domain from scheduling events with users of another domain.

In addition to access control, Calendar Server provides an additional level of security at the database protocol level for deployments that separate the calendar front end from the database back end. This level of security is referred to as Database Wire Protocol (DWP) authentication, and utilizes a user name / password pair to authenticate a DWP connection. The user name / password pairs on both the front end and database back end must be identical for a DWP connection to be authenticated.

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 can identify some of these attacks. Also, investigate unusual system load patterns and unusual connections. Review logs on a daily basis for any unusual activity.

Planning Calendar User Authentication

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

- Plain Text and Encrypted Password Login
- 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 Calendar Server administration. To understand directory server password policies, see the *Sun Java System Directory Server Deployment Planning Guide*:

`http://docs.sun.com/doc/817-7607`

Both plain text and encrypted password login can be used.

Certificate-based Authentication with Secure Sockets Layer (SSL)

Calendar Server uses the SSL protocol for encrypted communications and for certificate-based authentication of clients and servers. This section describes certificate-based SSL authentication.

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.

To authenticate with SSL, the calendar 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 Calendar 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 *Sun Java System Calendar Server Administration Guide*:

`http://docs.sun.com/doc/819-0024`

Chapter 19

Planning Calendar Server Services

This chapter describes the planning information for Calendar Server services.

This chapter contains the following sections:

- Planning for Calendar Server Front-end and Back-end Services
- Planning for the Calendar Server LDAP Data Cache

Planning for Calendar Server Front-end and Back-end Services

Calendar Server consists of six major services:

- HTTP Service (`cshttpd`) listens for HTTP requests. It receives user requests and returns data to the caller.
- Administration Service (`csadmind`) is required for each instance of Calendar Server. It provides a single point of authentication and administration for the Calendar Servers and provides most of the administration tools.
- Notification Service (`csnotify`) sends notifications of events and to-dos using either email or the Event Notification Service.
- Event Notification Service (`enpd`) acts as the broker for event alarms.
- Distributed Database Service (`csdwpd`) links multiple database servers together within the same Calendar Server system to form a distributed calendar store.

- Backup Service (`csstored`) implements automatic backups, both archival backups and hot backups. The first backup is a snapshot with log files, the second is a snapshot with log files applied. This service is automatically started when you run the `start-cal` command. However, it is not enabled at installation time, so you must configure it to function. If left unconfigured, Backup Service sends out a message to the administrator every 24 hours, with the notification that the service is not configured.

In a scalable Calendar Server deployment, you would deploy front-end systems in conjunction with a back-end server. The front-end systems would contain one instance of the `cshttpd` daemon per processor and a single Administration Service. A back-end server would contain an instance of Notification Service, Event Notification Service, Distributed Database Service and Administration Service.

Authentication and XML /XSLT transformation are two Calendar Service activities that generate heavy load. Additional CPUs can be added to meet quality of service requirements. In a scalable environment, these heavy load activities take place on the front-end system(s), permitting more CPUs to be added to individual front-end systems, or more front-end systems to be added, to meet quality of service requirements.

NOTE The preceding paragraph is not applicable if the Communications Express Calendar client is used for calendar access. Communications Express uses the WCAP protocol to access Calendar Server data and therefore the Calendar Server infrastructure is not doing the XML/XSLT translations. See Part V, "Deploying Communications Express" for information on deploying Communications Express.

Calendar back-end services usually require half the number of CPUs sized for the Calendar front-end services. To support quality of service by the Calendar front-end system, the Calendar back-end system should use around two-thirds of the front-end CPUs.

You will want to consider early on in a deployment separating the Calendar Service into front-end and back-end services. Assign a separate host name for the front-end services and back-end services so that when it comes time to separate the functionality onto different hosts, the changes are essentially internal and do not require users to change their methods of operation.

The Calendar Server HTTP process that is typically a component of the front-end services is a dominant user of CPU time. Account for peak calendar usage to provide enough front-end processing power to accommodate the expected peak HTTP sessions. Typically, you would make the Calendar Server front end more available through redundancy, that is, by deploying multiple front-end hosts. As the front-end systems do not maintain any persistent calendar data, they are not good candidates for HA solutions like Sun Cluster. Moreover, the additional hardware and administrative overhead of such solutions make deploying HA for Calendar Server front ends both expensive and time-consuming.

NOTE The only configuration for Calendar front ends that might warrant a true HA solution is where you have deployed the Calendar front end on the same host that contains a Messaging Server MTA. Even in this configuration, however, the overhead of such a solution should be carefully weighed against the slight benefit.

A good choice of hardware for the Calendar Server front ends is a single or dual processor server. You would deploy one instance of the Calendar Server `cshttpd` daemon per processor. Such a deployment affords a cost-effective solution, enabling you to start with some level of initial client concurrency capability and add client session capacity as you discover peak usage levels on your existing configuration.

When you deploy multiple front ends, a load balancer (with sticky/persistent connections) is necessary to distribute the load across the front-end services.

The Calendar Server back-end services are well balanced in resource consumption and show no evidence of bottleneck formation either in CPU or I/O (disk or network). Thus, a good choice of hardware for the back end would be a SPARC server with a single striped volume. Such a machine presents considerable capacity for large-peak calendar loads.

If your requirements include high availability, it makes sense to deploy the Calendar Server back end with Sun Cluster, as the back end does contain persistent data.

NOTE In a configuration with both front-end and back-end Calender Server hosts, all hosts must be running:

- The same operating system environment and version; that is, you cannot have a mixture of systems running Solaris SPARC, Solaris x86, Linux Red Hat, and so forth.
- The same releases of Calendar Server, including patch or hotfix releases.

Planning for the Calendar Server LDAP Data Cache

The LDAP data cache option ensures that LDAP data is available immediately after it has been committed. In some configurations of the LDAP directory server an update might need to be referred to a (remote) master server from which the update is then replicated down to the local LDAP directory. These kinds of configurations can induce a delay in the availability of committed data on the local LDAP server.

For example, if your site has deployed a master/slave LDAP configuration where Calendar Server accesses the master LDAP directory through a slave LDAP directory server, which in turn introduces a delay in the availability of committed LDAP data, the LDAP data cache can ensure that your Calendar Server clients have accurate LDAP data.

This section covers the following topics:

- Considerations for Using the LDAP Data Cache
- Master/Slave LDAP Configuration
- Resolving the Master-Slave Delay Problem
- Configuring the LDAP Data Cache

Considerations for Using the LDAP Data Cache

Use these guidelines to determine if your site should configure the LDAP data cache:

- If Calendar Server at your site accesses your master (or root) LDAP directory server directly with no delays in the availability of committed LDAP data, you don't need to configure the LDAP data cache. Make sure that the `local.ldap.cache.enable` parameter is set to "no" (which is the default).
- If your site has deployed a Master / Slave LDAP Configuration where Calendar Server accesses the master LDAP directory through a slave LDAP directory server, which in turn introduces a delay in the availability of committed LDAP data, configure the LDAP data cache to ensure that your end users have the most recent data.

Master/Slave LDAP Configuration

A Master / Slave LDAP configuration includes a master (root) directory server and one or more slave (consumer or replica) directory servers. Calendar Server can access the master LDAP directory server either directly or through a slave directory server:

- If Calendar Server accesses the master LDAP directory server directly, the LDAP should be accurate, and you do not need to configure the LDAP data cache.
- If Calendar Server accesses the master LDAP directory server through a slave directory server, LDAP data changes are usually written transparently via an LDAP referral to the master directory server, which in turn replicates the data back to each slave directory server.

In this second type of configuration, problems with inaccurate LDAP data can occur because of the delay in the availability of committed LDAP data to the slave directory servers.

For example, Calendar Server commits an LDAP data change, but the new data is not available for a specific amount of time because of the delay in the master directory server updating each slave directory server. A subsequent Calendar Server client operation uses the old LDAP data and presents an out-of-date view.

If the delay in updating the slave directory servers is short (only a few seconds), clients might not experience a problem. However, if the delay is longer (minutes or hours), clients will display inaccurate LDAP data for the length of the delay.

The following table lists the LDAP attributes that are affected by a delay in a master/slave LDAP server configuration where Calendar Server accesses the master LDAP directory server through a slave LDAP directory server.

Table 19-1 Calendar Server LDAP Attributes Affected by Delays

Operation	**LDAP Attributes Affected**
Auto provisioning	`icsCalendar, icsSubscribed, icsCalendarOwned, icsDWPHost`
Calendar groups	`icsSet`
Calendar creation	`icsCalendarOwned, icsSubscribed`
Calendar subscription	`icsSubscribed`
User options	`icsExtendedUserPrefs, icsFirstDay, icsTimeZone, icsFreeBusy`
Calendar searches	`icsCalendarOwned`

To ensure that your end uses have the most recent LDAP data, configure the LDAP data cache as described in the following section, "Resolving the Master-Slave Delay Problem."

Resolving the Master-Slave Delay Problem

The LDAP data cache resolves the master/slave LDAP configuration problem by providing Calendar Server clients with the most recent LDAP data, even when the master directory server has not updated each slave directory server.

If the LDAP data cache is enabled, Calendar Server writes committed LDAP data to the cache database (`ldapcache.db` file). By default, the LDAP cache database is located in the *cal_svr_base*`/var/opt/SUNWics5/csdb/ldap_cache` directory, but you can configure a different location if you prefer.

When a client makes a change to the LDAP data for a single user, Calendar Server writes the revised data to the LDAP cache database (as well as to the slave directory server). A subsequent client operation retrieves the LDAP data from the cache database. This data retrieval applies to the following operations for a single user:

- User's attributes at login
- User's options (such as color scheme or time zone)
- User's calendar groups

- User's subscribed list of calendars

Thus, the LDAP data cache database provides for:

- Data consistency across processes on a single system—The database is available to all Calendar Server processes on a multiprocessor system.
- Data persistence across user sessions—The database is permanent and does not require refreshing. You can configure the time to live (TTL) for an LDAP data cache entry and the interval between each database cleanup.

Limitations to the LDAP Data Cache

The LDAP data cache does not provide for:

- Reading the cache for searches where a list of entries is expected, for example, searching for attendees for a meeting. This type of search is subject to any LDAP delay. For instance, a newly created calendar will not appear in a calendar search if the LDAP search option is active and the search is performed within the delay period following the creation of a new calendar.
- Reading and writing of the cache across multiple front-end servers. Each front-end server has its own cache, which is not aware of data in other caches.
- The capability to handle a user who doesn't always log into the same server. Such a user will generate different LDAP data in the cache on each server.

Configuring the LDAP Data Cache

Configure the LDAP data cache by setting the appropriate parameters in the `ics.conf` file. See the *Sun Java System Calendar Server Administration Guide* for more information:

`http://docs.sun.com/doc/819-0024`

CAUTION If Calendar Server or the server where Calendar Server is running is not properly shut down, manually delete all files in the `ldap_cache` directory to avoid any database corruption that might cause problems during a subsequent restart.

Chapter 20

Understanding Calendar Server Pre-Installation Considerations

This chapter describes considerations you need to think about before installing Calendar Server.

This chapter contains the following sections:

- Calendar Server Installation Considerations
- Planning for Calendar Server Administrators
- Planning for Calendar Server Hosted Domains
- Post-Installation Calendar Server Configuration

Calendar Server Installation Considerations

The installation and configuration of Calendar Server has significantly changed from earlier Calendar Server releases (pre-2003Q4 versions). There is no longer a standalone installer for Calendar Server.

If you do not already have Calendar Server installed, you must use the Sun Java Enterprise System installer to get the 2005Q1 version. With this installer, you can also install other Sun Java System component products and packages. For information about the Java Enterprise System installer, refer to the *Sun Java Enterprise System 2005Q1 Installation Guide*:

`http://docs.sun.com/doc/819-0056`

If you want to upgrade from Calendar Server 6 2003Q4 to Calendar Server 6 2005Q1, the upgrade process is described in “Upgrading from Java Enterprise System 2003Q4” in the *Sun Java Enterprise System 2005Q1 Upgrade and Migration Guide*:

`http://docs.sun.com/819-0062`

For information about migrating from older versions of Calendar Server, up through version 5.x, see the Migrations Utility chapter in the *Sun Java Systems Calendar Server Administration Guide* (covers up to version 5.x):

`http://docs.sun.com/doc/819-0024`

For migrating from versions later than 5.x, contact your Sun support representative.

Which Calendar Server Components to Configure?

When you install Calendar Server software, the Java Enterprise System installer installs all the Calendar Server packages. You then configure the appropriate Calendar Server component on a Calendar host through the Calendar Server configurator program.

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

Table 20-1 Which Calendar Server Components to Configure?

Type of Calendar Host Being Configured	Needs These Components Selected in the Configurator Program
Front end	HTTP service(s) and Administration service
Back end	Notification Service, Event Notification Service, Distributed Database Service and Administration Service

The Distributed Database Service (`csdwpd`) is required only on back-end servers, that is, a server that has a calendar database, but does not provide user access services (`cshttpd`). It is not required on front-end servers that do not have a calendar database. The `csdwpd` service enables you to link front-end and back-end servers within the same Calendar Server configuration to form a distributed calendar store.

Planning for Calendar Server Administrators

Administrators for Calendar Server include:

- Calendar Server Administrator (calmaster)
- Calendar Server User and Group
- Superuser (root)

Calendar Server Administrator (calmaster)

The Calendar Server administrator is a specific user name with its associated password that can manage Calendar Server. For example, a Calendar Server administrator can start and stop Calendar Server services, add and delete users, create and delete calendars, and so on. This user has administrator privileges for Calendar Server but not necessarily for the directory server.

The default user ID for the Calendar Server administrator is `calmaster`, but you can specify a different user during Calendar Server configuration, if you prefer. After installation you can also specify a different user in the `service.admin.calmaster.userid` parameter in the `ics.conf` file.

The user ID you specify for the Calendar Server administrator must be a valid user account in your directory server. If the Calendar Server administrator user account does not exist in the directory server during configuration, the configuration program can create it for you.

See the *Sun Java System Calendar Serve Administration Guide* for the complete list of Calendar Server administrator configuration parameters in the `ics.conf` file:

`http://docs.sun.com/819-0024`

Calendar Server User and Group

On Solaris systems, these special accounts are the user ID and group ID under which Calendar Server runs. Use the default values, `icsuser` and `icsgroup`, which are automatically created by the configuration program, if they do not exist. If you prefer, however, you can specify values other than `icsuser` and `icsgroup` when you run the Calendar Server configuration program. These values are stored in the `local.serveruid` and `local.servergid` parameters, respectively, in the `ics.conf` file.

Superuser (root)

On machines running Solaris software, you must log in as or become `superuser` (`root`) to install Calendar Server. You can also run as `superuser` to manage Calendar Server using the command-line utilities. For some tasks, however, you should run as `icsuser` and `icsgroup` (or the values you have selected) rather than `superuser` to avoid access problems for Calendar Server files.

Planning for Calendar Server Hosted Domains

Calendar Server supports hosted (or virtual) domains. In a hosted domain installation, each domain shares the same instance of Calendar Server, which enables multiple domains to exist on a single server. Each domain defines a name space within which all users, groups, and resources are unique. Each domain also has a set of attributes and preferences that you specifically set.

When installing and configuring hosted domains, use Schema 2 only.

Installing and configuring hosted domains on a server involves these high-level steps:

1. Installing and configuring Directory Server
2. Installing and configuring Web Server or Application Server
3. Installing and configuring Access Manager

   The Delegated Administrator is installed with Access Manager.
4. Installing Calendar Server
5. Running the `comm_dssetup.pl` script

   For instructions on running this script, see Chapter 2 in the *Sun Java System Calendar Server Administration Guide:*

   `http://docs.sun.com/doc/819-0024`
6. Configuring Communications Services Delegated Administrator

   For instructions on configuring and using the Communications Services Delegated Administrator utility, see the *Sun Java System Communications Services Delegated Administrator Guide*:

   `http://docs.sun.com/doc/819-0114`

7. Creating default domain and site administrator (`calmaster`)

   The default domain is created when `commadmin` is configured, but the domain entry must be modified to add Calendar (or Mail) services. Also, the site calendar administrator (`calmaster`) must be set up. For instructions on how to perform these two tasks, see "Post Configuration Tasks" in the *Sun Java System Calendar Server Administration Guide.*

8. Configuring Calendar Server

   For instructions on running the `csconfiguratior.sh` program, see Chapter 3 in the *Sun Java System Calendar Server Administration Guide*.

9. Setting hosted domain configuration parameters for Calendar Server

   For a list of the configuration parameters and their values, see "Hosted Domain Configuration Parameters" in the *Sun Java System Calendar Server Administration Guide*.

10. Creating the hosted domains for your site by using the `commadmin` utility

11. Populating your hosted domains with users and resources by using the `commadmin` utility

12. Starting Calendar Server services

    For instructions, see the *Sun Java System Calendar Server Administration Guide*.

NOTE Always perform your provisioning for Schema 2 with the Communications Services Delegated Administrator interface.

Schema 1 provisioning tools do not support hosted domains.

Post-Installation Calendar Server Configuration

After you install the Calendar Server software, you must configure it. This step was previously performed as part of the installation process, but has now been separated out of the installer.

After you install Calendar Server, you must configure Calendar Server as follows:

1. Run the Directory Server Setup script (`comm_dssetup.pl`) to configure Sun Java System Directory Server.

2. Run the Calendar Server configuration program (`csconfigurator.sh`) to configure your site's specific requirements and to create a new `ics.conf` configuration file. For a description of the parameters in the `ics.conf` file, see the *Sun Java System Calendar Server Administration Guide*:

    `http://docs.sun.com/doc/819-0024`

The `comm_dssetup.pl` script is located in the `/opt/SUNWcomds/sbin` directory, and the `csconfigurator.sh` utility is located in the `/opt/SUNWics5/cal/sbin` directory.

There are some configuration settings and changes that the Java Enterprise System installer and Calendar Server configuration utility (`csconfigurator.sh`) do not make. You must manually make changes to the following items:

- **DWP and CLD Configurations.** Edit the `ics.conf` file so that the CLD cache option is enabled. This cache stores the DWP host server information for calendar users and thus reduces calls to the LDAP directory server.
- **Default Time Zone.** If your default time zone is not Americas/New York, change it by editing the `ics.conf` file. You also need to change it in the `/opt/SUNWics5/cal/bin/html/default_user_prefs.xml` file so that it is in sync with the `ics.conf` file.
- **Client-side Rendering.** Calendar Server performs client-side rendering by downloading the XSLT processing to the end user's browser, which in turn reduces the processing that must be done by Calendar Server. Calendar Server downloads the XSLT processing only if the browser is capable of rendering the XSLT processing. In the current release, this applies only to Internet Explorer 6.0. Edit the `ics.conf` file to make this performance improvement to client-side rendering.
- **Setting for `tmpfs`.** Edit the `tmpfs` setting for performance enhancement.

For more information on these changes, see the *Sun Java System Calendar Server Administration Guide*.

Part IV

Deploying Instant Messaging

Chapter 21

Introduction to Instant Messaging Software

Instant Messaging enables secure, real-time communication and collaboration, combining presence awareness with instant messaging capabilities such as chat, conferences, alerts, news, polls, and file transfers to create a rich collaborative environment. These features enable one-to-one as well as group collaboration through either short-lived communications or persistent venues such as conference rooms or news channels.

Instant Messaging ensures the integrity of communications through its multiple authentication mechanisms and Secure Sockets Layer (SSL) connections. Integration with Portal Server and Access Manager brings additional security features, services-based provisioning access policy, user management, and secure remote access.

This chapter contains the following sections:

What Is an Instant Messaging Service?

At a simplistic level, an instant messaging service:

- Accepts instant messages from external sites
- Determines the user to which the message should be delivered, and routes it accordingly
- Accepts instant messages from internal hosts
- Determines the destination system to which the message should be delivered, and routes it accordingly

In addition, an instant messaging service can provide real-time conferencing, news and calendar alerts, and for offline users, email message forwarding.

Of crucial importance to a good instant messaging service is that the service provides for scalability, high availability, reliability, and good performance.

Instant Messaging Core Product Components

Instant Messaging contains the following core components:

- **Instant Messenger Resources (client)**. A set of files that makes up the client program for end users to initiate, compose, and reply to messages. Typically users also use the client to participate in conferences. The client is also called Sun Java System Instant Messenger.
- **Instant Messaging Server.** An electronic message delivery system that supports instant message delivery from one system to another system. The server serves the presence information to Instant Messenger clients, enables end users to establish sessions, and enforces policies.
- **Instant Messaging Multiplexor.** A scalability component that consolidates messenger connections. To support large deployments, with thousands of concurrent connections, Instant Messaging uses a connection multiplexor to improve server scalability. This component opens a single connection to the Instant Messaging server. In addition to scalability, you can install the multiplexor outside the firewall while leaving the server inside the firewall to protect it from unauthorized external access. The Instant Messaging multiplexor is also referred to as the multiplexor.

- **Access, Communication, and Transfer Protocols.** These protocols, such as LDAP, HTTP, TCP/IP, and SMTP, can be found in "Instant Messaging Supported Standards" on page 286.
- **Access Manager Instant Messaging Service Definition.** Instant Messaging provides a service definition to Access Manager using the Access Manager SDK, to provide support for Access Manager managed policies and SSO capabilities.
- **Instant Messaging API.** Enables you to create custom Instant Messaging clients.

Components Related to Instant Messaging

The software components discussed in this section work with Instant Messaging server, but are installed separately. Chapter 23, "Developing an Instant Messaging Architecture" provides more detailed information that illustrates how these servers interact with Instant Messaging.

Web Server

(Required) For any deployment, you need to install a web server, such as Sun Java System Web Server or Sun Java System Application Server. You can also use any open standard web server (for example, Apache). In all cases, the Instant Messenger resources must reside on the web server host.

Instant Messaging requires a web server to serve the Instant Messenger resources. The Instant Messenger resource files include:

- The `index.html` file, provided by Instant Messenger, or a home page with a link to invoke Instant Messenger
- Instant Messenger jar files (`messenger.jar`, `imres.jar`, `imbrand.jar`, `imdesktop.jar`, `imnet.jar`, and `imjni.jar`)
- The Instant Messenger Online Help

You must install Instant Messenger resources on the same host where the web server is installed. In an Access Manager deployment, you can install these resources on the Access Manager server's host or on a different web server host. In most cases, the resources will be installed on the same host where you installed the Instant Messaging server software. It is possible to locate the Instant Messenger resources on a host other than the Instant Messaging server or multiplexor.

NOTE Install the web server before configuring Instant Messaging.

LDAP Server

(Required) Instant Messaging uses an LDAP server, such as Directory Server, for end user authentication and search. In a deployment with Portal Server, Instant Messaging uses the same LDAP server used by Portal Server. If you do not have an LDAP directory already installed, you must install one.

The Instant Messaging server does not store the Instant Messenger end-user authentication information. This information is stored in the LDAP server.

By default, the Instant Messaging server relies on the common end-user attributes `cn` and `uid` to search for end-user and group information. If you want, you can configure the server to use another attribute for search. In addition, Instant Messaging properties (such as contact lists and subscriptions) can be stored in files on the Instant Messaging server or in the LDAP server.

For instructions on configuring the server to use a non-default attribute for user search, see the *Sun Java System Instant Messaging Administration Guide*:

http://docs.sun.com/doc/819-0430

NOTE Because a proper Directory Server implementation is instrumental to a successful Instant Messaging deployment, read the *Sun Java System Directory Server Deployment Guide* in addition to this guide:

http://docs.sun.com/doc/817-7607

SMTP Server

(Optional) An SMTP messaging server, such as Messaging Server, is used to forward instant messages, in the form of email, to end users who are offline. The SMTP server does not have to reside on the same host as the Instant Messaging server.

Calendar Server

(Optional) Calendar Server is used to notify users of calendar-based events.

Access Manager and Access Manager SDK

(Optional) Access Manager and Access Manager SDK provide end user and service management, authentication and single sign-on services. In addition, Access Manager and Access Manager SDK are required in deployments that include Portal Server. In both deployments, the SDK must be installed on the Instant Messaging server's host.

Portal Server

(Optional) Portal Server supports message archiving, and enables you to run Instant Messaging in secure mode. In addition, the Instant Messenger client is made available to end users through the Portal Server desktop. The following two Portal Server components provide additional functionality:

- Portal Server Desktop
- Secure Remote Access

Portal Server Desktop

Instant Messenger installed in the Portal Server environment can be launched from the Instant Messaging channel available to end users on Portal Server Desktop.

Secure Remote Access

Secure Remote Access enables remote end users to securely access their organization's network and its services over the Internet. The end user can access Secure Remote Access by logging into the web-based Portal Server Desktop through the portal gateway. The authentication module configured for Portal Server authenticates the end user. The end-user session is established with Portal Server and the access is enabled to the end user's Portal Server Desktop.

In the Portal Server environment, you can configure Instant Messenger in either secure or non-secure mode. In secure mode, communication is encrypted through the Portal Server Netlet. When access Instant Messenger in secure mode, a lock icon appears in the Status area of the Instant Messenger application. In non-secure mode, the Instant Messenger session is not encrypted. For more information on Netlet, see the *Sun Java System Portal Server Secure Remote Access Administration Guide*:

`http://docs.sun.com/doc/817-5317`

Instant Messaging Supported Standards

Instant Messaging is built on native Internet technology, so you can maintain a single architecture inside and outside your organization, even when collaborating with your customers and partners. Additionally, you aren't locked into a proprietary system. All key components of Instant Messaging are based on proven, open Internet standards such as:

- **LDAP.** Provides access to enterprise directory information, enabling an accurate, secure instant messaging system.
- **HTML.** Formatting language for providing web browser access to the client.
- **HTTP.** HypterText Transport Protocol for providing web browser access to the client.
- **SMTP.** Simple Mail Transfer Protocol for reliable delivery of instant messages over Internet mail messages.
- **TCP/IP.** Proven, worldwide networking protocol.
- **XMPP.** Extensible Message and Presence Protocol for interoperating with public networks through open source gateways.

Instant Message Structure Format

XMPP protocol is used to format the instant messages. The message bodies themselves may be wrapped in HTML.

Access Protocol

In Instant Messaging, user information and preferences are retrieved from an LDAP directory. This directory can either be dedicated for use by Instant Messaging, or be shared by other components such as Access Manager or Portal Server. User data is typically retrieved using LDAP search functions. Instant Messaging deployments that make use of Access Manager and Portal Server make use of the same LDAP server.

Communication and Message Transfer Protocols

Instant Messaging server-to-server and client-to-server communications occur over TCP/IP.

Instant Messaging uses SMTP to send messages to offline users.

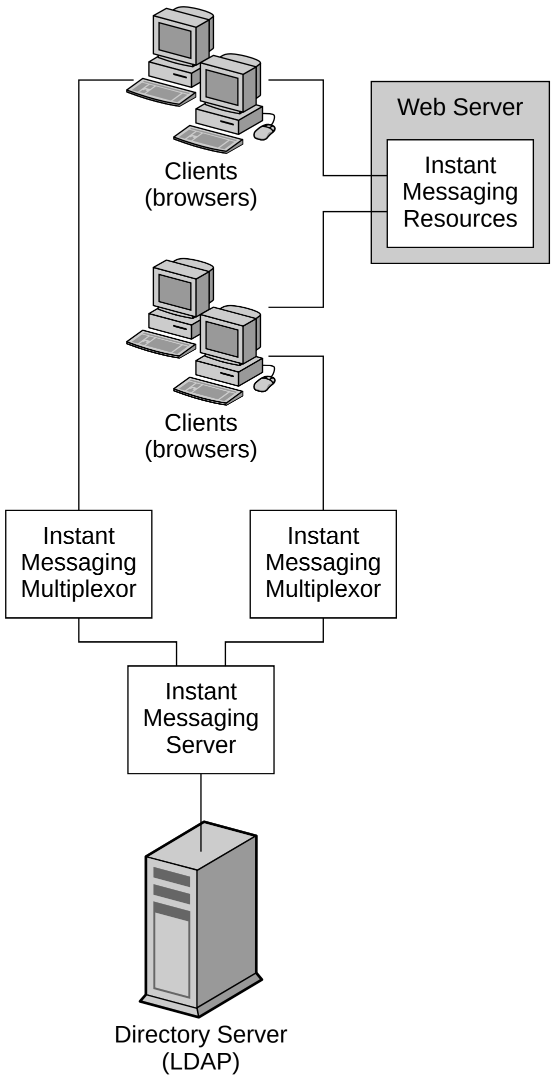
Browsers use HTTP to retrieve Instant Messenger resource files from the Web server. Once retrieved, the browser reads the HTML and displays the contents of the files.

Instant Messaging 7 is an XMPP / Jabber client / server solution, able to communicate with XMPP-compliant servers, clients, and gateways. Gateways are available in the open-source community to enable communication between Jabber and AOL, Yahoo, and other instant messaging systems.

Instant Messaging Software Architecture

Figure 21-1 on page 288 shows the Instant Messaging software architecture.

Figure 21-1 Instant Messaging Software Architecture



The web server (or an application server with a web service embedded), downloads the Instant Messaging resources from a browser to the clients. The resource files make up the client. Clients send messages to one another through a multiplexor, which forwards the messages to the Instant Messaging server.

The directory server stores and retrieves local user and group delivery information such as preferences, location, and to which multiplexor to route messages for this user. When the Instant Messaging server receives a message, it uses this information to determine where and how the message should be delivered. In addition, the directory server can contain user information such as contact lists and subscriptions.

In this basic configuration, Instant Messaging directly accesses Directory Server to verify user login name and passwords for mail clients that use Instant Messaging.

Outgoing instant messages from clients go directly to the multiplexor. The multiplexor sends the message to the appropriate Instant Messaging server, which in turn forwards the message to another Instant Messaging server, or if the message is local, to the multiplexor with which the recipient is associated. (See "Instant Messaging Physical Deployment Examples" on page 327 for illustrations of this process.)

New users are created by adding user entries to the directory. Entries in the directory can be created or changed by using the tools provided with the directory server.

Instant Messaging components are administered through a set of command-line interfaces and text-based configuration files. Any machine connected to the Instant Messaging host can perform administrative tasks (assuming, of course, the administrator has the required privileges).

NOTE Typical Instant Messaging deployments are not installed on a single machine. They also have additional features like multiplexing and high availability enabled. See Chapter 23, "Developing an Instant Messaging Architecture" for more information.

The following sections outline the three primary components of Instant Messaging in further detail:

- Instant Messaging Server
- Instant Messaging Multiplexor
- Instant Messenger Client

Instant Messaging Server

The Instant Messaging server handles tasks such as controlling Instant Messenger privileges and security, enabling Instant Messenger clients to communicate with each other by sending alerts, initiating chat conversations, and posting messages to the available news channels. The Instant Messaging server also handles archiving, calendar alerts, and offline email notifications

The Instant Messaging server supports the connection of a multiplexor that consolidates connections over one socket. For more information on the multiplexor, see "Instant Messaging Multiplexor."

Access control files and Access Manager policies are used for administration of end users, news channels, and conference rooms.

The Instant Messaging server routes, transfers, and delivers instant messages for the Instant Messaging product.

Direct LDAP Lookup

The server can look up directory information directly from the LDAP server. The results of the LDAP queries are cached in the process, with configurable aging and expiration, so settings are tunable. Refer to the *Sun Java System Directory Server Administration Guide* for further information:

`http://docs.sun.com/doc/817-7613`

Message Delivery

After the message is processed, the server sends the message to the next stop along the message's delivery path. This can be the intended recipient's multiplexor or another server. Once received by a multiplexor, the message is routed directly to the intended recipient. See "Basic Instant Messaging Architecture" on page 312 for a illustration of this process.

Instant Messaging Multiplexor

The Instant Messaging multiplexor component connects multiple instant messenger connections into one TCP (Transmission Control Protocol) connection, which is then connected to the Instant Messaging server. The multiplexor reads data from Instant Messenger and writes it to the server. Conversely, when the server sends data to Instant Messenger, the multiplexor reads the data and writes it to the appropriate connection. The multiplexor does not perform any end user authentication or parse the client-server protocol (IM protocol). Each multiplexor is connected to one and only one Instant Messaging server.

You do not need to install the Instant Messaging multiplexor, that is, you can configure Instant Messaging without a multiplexor. However, production deployments should be configured to use the multiplexor.

You can install multiple multiplexors based on your deployment requirements. For more information, see Chapter 23, “Developing an Instant Messaging Architecture.”

Instant Messenger Client

Instant Messenger is Instant Messaging’s client that can be configured to be a browser-based applet using Java plugin, or a standalone Java application using Java™ Web Start.

To run Instant Messenger client on Solaris or Linux, you must use Java Web Start. On Microsoft Windows you can run Instant Messenger as an applet or a Java Web Start application. In most cases, run Instant Messenger as a Java Web Start application.

For more information on customizing Instant Messenger, see the *Sun Java System Instant Messaging Administration Guide*:

`http://docs.sun.com/doc/819-0430`

Instant Messenger provides the following modes of communication:

- **Chat.** Instant Messenger’s version of Instant Messaging conferences is called chat. Chat is a real-time conversation capability that enables end users to complete projects, answer customer queries, and complete other time-critical assignments. Chat sessions (two or more participants) are held in chat rooms created on a need basis.
- **Conference Rooms.** Conference rooms are persistent chat rooms that work similarly to regular chat sessions, but offer:
  - Access control
  - Moderated chats

- **Alerts.** Alerts enable information delivery and response to end users through the Instant Messenger interface. Alerts can deliver time-critical information to the end user. The sender of the alert message is notified when the message is delivered and read by the recipient. You can also configure Instant Messaging to forward alerts to an email address.
- **Poll.** The polling function enables you to ask end users for their response to a question. You can send a question and possible answers to poll recipients, and the recipients can respond with their selected answer.
- **News.** News channels are forums for posting and sharing information. End users can subscribe to news channels of interest to see updates using the URL of the news channels or view the news channel updates through static messages. Administrators control news channel access by assigning end users to the channels they need, and deciding who can see or post information to the channels.

NOTE Instant messages can contain embedded URLs. If you are using proxy servers, you might need to have clients using Java Web Start modify their proxy configuration for resolving such URLs.

For more information on configuring the proxy settings manually, see the *Sun Java System Instant Messaging Administration Guide*:

`http://docs.sun.com/doc/819-0430`

Designing Your Instant Messaging Deployment

The deployment process consists of the following general phases, referred to as the Solution Life Cycle:

- Analyzing business requirements
- Analyzing technical requirements
- Designing the logical architecture
- Designing the deployment architecture
- Implementing the deployment
- Operating the deployment

The deployment phases are not rigid: the deployment process is iterative in nature.

For detailed information on the deployment process for Instant Messaging, and other Java Enterprise System components, see the *Sun Java System Deployment Planning Guide*:

`http://docs.sun.com/doc/819-0058`

Chapter 22

Planning an Instant Messaging Sizing Strategy

This chapter introduces the concepts, context, and rationale of sizing your Instant Messaging deployment.

This chapter contains the following sections:

Instant Messaging Sizing Strategy Overview

When you design your deployment, you must decide how to configure your Instant Messaging server to provide optimum performance, scalability, and reliability.

Sizing is an important part of the design 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 Instant Messaging server users generate. Sizing is an iterative effort.

Because each deployment has its own set of unique features, this chapter will not provide detailed Instant Messaging sizing information for your specific site. Also, this chapter will not provide sizing information for servers with which Instant Messaging interoperates, such as LDAP, SMTP, and so on. Rather, this chapter explains what you need to consider when you architect your sizing plan. In addition, you'll find general guidelines for Instant Messaging components you can modify to suit your site's needs. Work with your Sun technical representative for your deployment hardware and software needs.

Collecting Instant Messaging Sizing Data

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

- Determining Peak Volume of Unique Instant Messaging Logins
- Creating Your Instant Messaging Usage Profile
- Defining Your Instant Messaging User Base or Site Profile

Determining Peak Volume of Unique Instant Messaging Logins

Your *peak volume* is the largest concentrated numbers of unique logins to your Instant 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 groups might occur during corporate-held core hours which differ between time zones.

Analyzing peak volume involves three basic operations:

1. Determining when and for how long the peaks occur
2. Sizing your deployment against peak volume load assumptions

   Once patterns are analyzed, choices can be made to help the system handle the load and provide the services that users demand.
3. Making sure that your Instant Messaging deployment can support the peak volume that you have determined

Creating Your Instant Messaging Usage Profile

Measuring your load is important for accurate sizing. Your *usage profile* determines the factors that programs and processes place on your Instant Messaging servers and multiplexors.

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

Creating a usage profile involves answering the following questions:

1. What is the total number of users on your system?

When counting the number of users on your system, account for not only the users who have accounts and can log into the system, but also the users with accounts who are currently not logged into the system. The following table describes the types of users that make up the total.

Table 22-1 Instant Messaging Active Versus Inactive User

Connection	Description
Inactive User	A user with an Instant Messaging account who currently is not logged into the system. Non-connected users consume disk space but no CPU or memory.
Connected/Inactive	These users are logged in, but are not currently sending or receiving instant messages.
Connected/Active	Logged into the system and actively sending messages, updating user information such as contact lists, and attending conferences throughout the day.

Characterize your configured users using these three general profiles. The total of these users should give you an idea of the total number of concurrent connections you need to support.

If you have a small deployment, the defaults might be sufficient to meet your site's needs. So, if you have a very small deployment (for example, less than 300 users), you might not need to go through this process of planning a sizing strategy. Work with your Sun Client Services representative to determine your individual needs.

2. How many connections are on your system during your peak volume?

   Correctly formulating the maximum number of concurrent users that has to be sustained by the system is key to planning your resource requirements. Although a deployment usually has maximum number of configured users, it is important to plan for the maximum number of concurrent users (connected and more or less active). A conservative estimate for the number of concurrent users can then be determined based on a 1:10 ratio. Thus, for a deployment of 50,000 configured users, the concurrent users would be 5,000.

   Specifically, note the number of concurrent connections, idle, and busy connections.

Table 22-2 Instant Messaging Client Connections

Connection	Description
Concurrent Connection	Number of unique TCP connections or sessions that are established on your system at any given time.
Idle Connection	Connection where no information is being sent between the client and multiplexor or server and multiplexor.
Busy Connection	A connection that is in progress. An established connection where information is being sent between the client and multiplexor or multiplexor and server.

   To determine the number of *concurrent connections* in your deployment, you can count the number of established TCP connections by using the `netstat` command on Solaris platforms.

   To determine the number of concurrent connections you can support, you need to obtain two values from parameters within the `iim.conf` file that are used for tuning multiplexor performance:

   a. `iim_mux.numinstances` - Specifies the number of multiplexor instances.

   b. `iim_mux.maxsessions` - Specifies the maximum number of clients that one mutliplexor process can handle. The default is 1000.

   Once you have obtained these values, multiply the `numinstances` number by the `maxsessions` number. This gives the total number of concurrent connections supported by your deployment. For information on the `iim.conf` file, see the *Sun Java System Instant Messaging Administration Guide*:

   `http://docs.sun.com/doc/819-0430`

3. If you have a large deployment, how will you organize your users?

   For example, consider placing active users on one server and inactive users on another.

4. What is the amount of storage used for each user?

   If you are not storing your end user data, such as contact lists, in LDAP, you need to plan for the space required to store this data. If you configure the server to store this data outside LDAP, the server stores it in a flat file. See the *Sun Java System Instant Messaging Administration Guide* for more information:

   `http://docs.sun.com/doc/819-0430`

5. How many messages enter your Instant 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 system?

   ❍ The Internet?

   This number of messages is also measured in messages per second during the peak volume.

7. Will you be using SSL? If yes, what percentage of users and what type of users?

   For example, in a particular organization, 20% of connections during peak hours will enable SSL.

Answering these questions provides a preliminary usage profile for your deployment. You can refine your usage profile as your Instant Messaging 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, do you need to consider high availability.

2. What backup and restore strategy do you have in place (such as disaster recovery and site failover)? What are the expected times to accomplish recovery tasks?

   Typically you need to back up the server configuration files, database, and any resource files you have customized.

Defining Your Instant Messaging User Base or Site Profile

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 Instant Messaging operations that your users will perform. Instant Messaging users fall into one of these user bases:

- Casual Users
- Heavy Users

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 an Instant Messaging Load Simulator" on page 301).

Casual Users

A lightweight user base typically consists of users with simple Instant Messaging requirements. These users rarely initiate chat sessions and rarely receive invitations. They might only use Instant Messaging as a presence tool.

Heavy Users

A heavy user uses significantly more system resources than a casual user. Typical usage for a this type of user may be something like the following:

- Presence updates equal to or greater than 20 times a day.
- Contact list contains about 30 contacts.
- Subscribes to the presence updates of all the contacts in the contact list.
- Sets up around four conferences or chats per day where each conference has three people in the conference room and lasts an average of 10 minutes, and a message is added to the conference every 1-15 seconds.

Using an Instant Messaging Load Simulator

To measure the performance of your Instant Messaging architecture, use your user bases (described in "Defining Your Instant Messaging User Base or Site Profile" on page 300) and your usage profile (described in "Creating Your Instant Messaging Usage Profile" on page 297) 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. Using a load simulator involves five basic steps:

1. Defining the user base that you want to test (for example, casual users)

   If necessary, adjust individual parameters to best match your usage profile.

2. Defining the hardware that will be tested
3. Running the load simulator and measuring the maximum number of concurrent connections on the tested hardware with the user base
4. Publishing your results and comparing those results with production deployments
5. Repeating 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 Client Services for recommended load simulators and support.

Understanding Instant Messaging System Performance Guidelines

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

Instant Messaging 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 Instant Messaging server to operate at peak volume. With sufficient memory, Instant Messaging can operate efficiently without excessive swapping.

For most deployments, you need at least 256 MB of RAM. The amount of RAM needed depends on the number of concurrent client connections, and whether the server and multiplexor are deployed on the same host. For information about concurrent connections, see "Creating Your Instant Messaging Usage Profile" on page 297. For information on hosting the server and multiplexor on the same host, see "Developing Instant Messaging Architectural Strategies" on page 305.

On Solaris systems, you can set the amount of memory allocated to the server by modifying the `iim.jvm.maxmemorysize` parameter in the `iim.conf` file. This parameter specifies the maximum number of megabytes of memory that the Java Virtual Machine (JVM) running the server is allowed to use. The default setting is 256 MB, and the maximum setting is 500 MB. For instructions on modifying this parameter, see the *Sun Java System Instant Messaging Administration Guide*:

`http://docs.sun.com/doc/819-0430`

You cannot currently change this value on Windows NT systems.

Instant Messaging 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 Instant Messaging. To improve efficiency in your system's disk throughput:

- Consider your maintenance operations, and ensure you have enough bandwidth for backup. Backups can also affect network bandwidth, particularly remote backups. Private backup networks can be a more efficient alternative.
- Carefully partition the data stores to improve throughput efficiency.
- Ensure the user base is distributed across RAID (Redundant Array of Independent Disks) environments in large deployments. Typically, you make this decision as part of the Directory Server deployment planning process.
- Stripe data across multiple disk spindles in order to speed up operations that retrieve data from disk.

Instant Messaging Disk Capacity

When planning Instant Messaging server system disk space, be sure to include space for operating environment software, Instant Messaging software, and for any servers not currently in your network that need to be installed to support Instant Messaging (such as LDAP). Be sure to use an external disk array. In addition, user disk space needs to be allocated. Typically, this space is determined by your site's policy. Typical installations will require:

- Approximately 300 MB of free disk space for each server or multiplexor
- Approximately 5 K of disk space for each user
- Additional space for the Instant Messaging archive

  The archive captures instant messages and archives them in a Portal Server Search database. End users can query and retrieve these archived messages using the Search page on the Portal Server desktop.

Use the following table to determine server and multiplexor disk space sizing numbers whether archiving is enabled or disabled. The figures listed in the table were generated using a 400MHz Ultra SPARC II Processor.

Table 22-3 Instant Messaging Server and Multiplexor Memory Disk Space Sizing for Concurrent Users

	Server Memory Consumption for Connected/Inactive Users	**Server Memory Consumption for Connected/Active Users**	**Multiplexor Memory Consumption for Connected/Inactive Users**	**Multiplexor Memory Consumption for Connected/Active Users**
Archive Disabled	8 MB +20 K per User	120 MB + 20 K per User	8 MB + 20 K per User	8MB + 28K per User
SSO/Portal/ Archive enabled	100MB +25K per User	120MB +30K per User	8M+35K per user	8 MB +40K per user

Instant Messaging 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.

Improving data integrity, system performance, and network congestion reduces the number of retransmissions. Steps to do this involve:

- Avoiding bottlenecks to ensure that the network infrastructure can handle the load
- Partitioning your network
- Not using theoretical maximum values when configuring your network to ensure that sufficient capacity exists for future expansion
- Separating traffic flows on different network partitions to reduce collisions and to optimize bandwidth use.

Instant Messaging CPU Resources

Enable enough CPU for your servers and multiplexing services. In addition, enable enough CPU for any RAID systems that you plan to use. If you intend to use archiving in your deployment, you need to take those space requirements into consideration as well.

Use the following table to help determine the number of CPUs your installation requires for optimum performance whether archive is enabled or disabled. The figures listed in the table were generated using a 400MHz Ultra SPARC II Processor.

Table 22-4 Instant Messaging CPU Utilization Numbers

	Server CPU Utilization for Connected/Inactive Users	**Server CPU Utilization for Connected/Active Users**	**Multiplexor CPU Utilization for Connected/Inactive Users**	**Multiplexor CPU Utilization for Connected/Active Users**
Archive Disabled	Several hundred thousand users per CPU	30 K users per CPU	50 K users per CPU	5 K users per CPU

Instant Messaging Multiplexor Configuration Best Practices

Consider the following suggestions and generalizations when planning your multiplexor deployment. The parameters discussed in this section are located in the `iim.conf` file.

- The number of `iim_mux.maxthreads` should not exceed the number of CPUs on your server.

This helps maximize resource utilization and optimizes processing speed.

- The `iim_mux.maxsessions` value should be high enough to avoid rejecting connections, but it should be reasonable enough so that the multiplexor processes to not get overloaded.
- Be sure that your expected number of concurrent client connections is less than the maximum possible by a safe margin.
- Do not configure threads or number of concurrent sessions to more than you require. Otherwise, you will unnecessarily consume system resources.
- A good starting point is to configure `iim_mux.numinstances` to the number of CPUs on the system.

See the *Sun Java System Instant Messaging Administration Guide* for more detailed information about these parameters:

`http://docs.sun.com/doc/819-0430`

Developing Instant Messaging Architectural Strategies

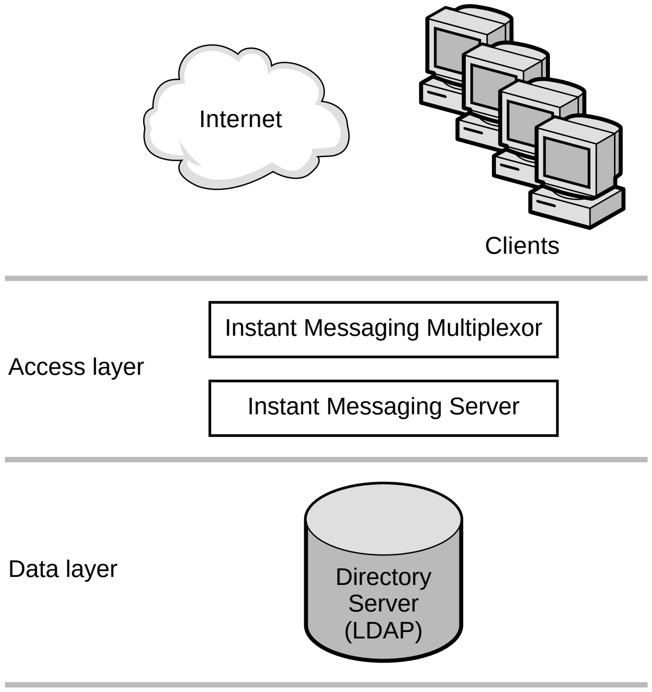
Once you have identified your system performance needs, the next step in sizing your Instant Messaging deployment is to size specific components based on your architectural decisions.

The topics in this section point out sizing considerations when you deploy two-tiered and one-tiered architectures. Using load balancers with Instant Messaging is also discussed.

Two-tiered Instant Messaging Architecture

A two-tiered architecture splits the Instant Messaging server deployment into two layers: an access layer and a data layer. In a simplified two-tiered deployment, you might add one or more multiplexors and servers to the access layer. The multiplexor acts as a proxy for users, and the relays messages to the Instant Messaging server. The data layer holds the Instant Messaging server database and Directory servers. Figure 22-1 on page 306 shows a simplified two-tiered Instant Messaging architecture.

Figure 22-1 Simplified Two-tiered Instant Messaging Architecture



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

- Are easier to maintain than one-tiered architectures
- Enable the offloading of load-intensive processes like SSL, message reprocessing
- Are easier for growth management and you can upgrade your system with limited overall downtime

Sizing Your Multiplexing Services

When you size your multiplexor, the calculation is based on your system load, particularly the number of concurrent connections the multiplexor needs to handle.

In addition, you must:

1. Add CPU or a hardware accelerator for SSL if appropriate.
2. Add memory to the machine if the multiplexor is being configured on it.

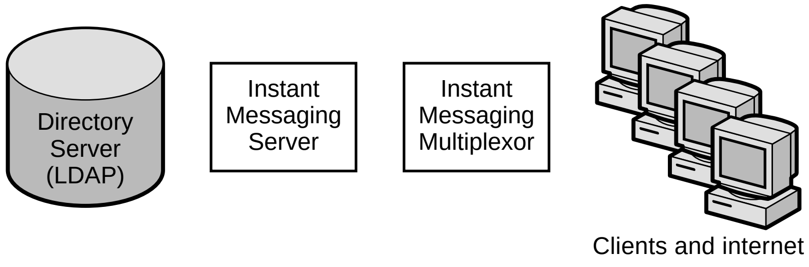
3. Account for denial of service.
4. Add capacity for load balancing and redundancy, if appropriate.

   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-tiered Instant Messaging Architecture

In a one-tiered architecture, there is no separation between access and data layers. The Instant Messaging server, multiplexor, and sometimes the Directory server are installed in one layer. The following figure illustrates the idea.

Figure 22-2 Simplified One-tiered Instant Messaging Architecture



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

To size a one-tiered architecture:

1. Add CPU for SSL, if necessary.
2. Account for denial of service attacks.
3. Add more disks for the increased number of client connections.
4. Add more disks for each multiplexor.

For specific instructions on sizing Instant Messaging components in one-tiered or two-tiered architectures, contact your Sun Client Services representative.

Using Load Balancers With Instant Messaging

Instant Messaging supports the use of load balancers located in front of the Instant Messaging multiplexors. However, you cannot currently use load balancers between the Instant Messaging multiplexors and the Instant Messaging server.

When deploying Instant Messaging as part of a Portal Server / Secure Remote Access deployment, you can locate load balancers between the Secure Remote Access gateway and the Instant Messaging multiplexors.

NOTE If you just need security for the client connection, and not HTTP tunneling, consider using SSL instead of Secure Remote Access. You can configure a secure Instant Messaging client connection by enabling SSL on the multiplexors and locating them outside the firewall.

Example Instant Messaging Resource Requirements

This section provides example resource distributions and recommended sizing information for the following two Instant Messaging deployment types:

- Small Deployment Sample Resource Requirements Numbers
- Large Deployment Sample Resource Requirements Numbers

Small Deployment Sample Resource Requirements Numbers

For a small Instant Messaging deployment with the server and multiplexor on a single server having 10,000 users with the following profile:

- 30 percent connected / active
- 20 percent connected / inactive
- 50 percent not connected

The memory requirements are: one or two CPUs with 300-500 MB RAM each.

Large Deployment Sample Resource Requirements Numbers

For a large Instant Messaging deployment having 1,000,000 users with the following profile:

- 5 percent connected / active
- 20 percent connected / inactive
- 75 percent not connected

The server memory requirements are 4 GB RAM on two CPUs. The multiplexor requirement is 4 GB RAM on 16 CPUs.

Chapter 23

Developing an Instant Messaging Architecture

This chapter describes a variety of Instant Messaging architectures. Depending on your deployment, you will need to install different components. For example, to support email notification, you need to install an SMTP server. If you do not want to support email notification, do not install an SMTP server.

For more detailed information about components that interoperate with Instant Messaging, see "Components Related to Instant Messaging" on page 283.

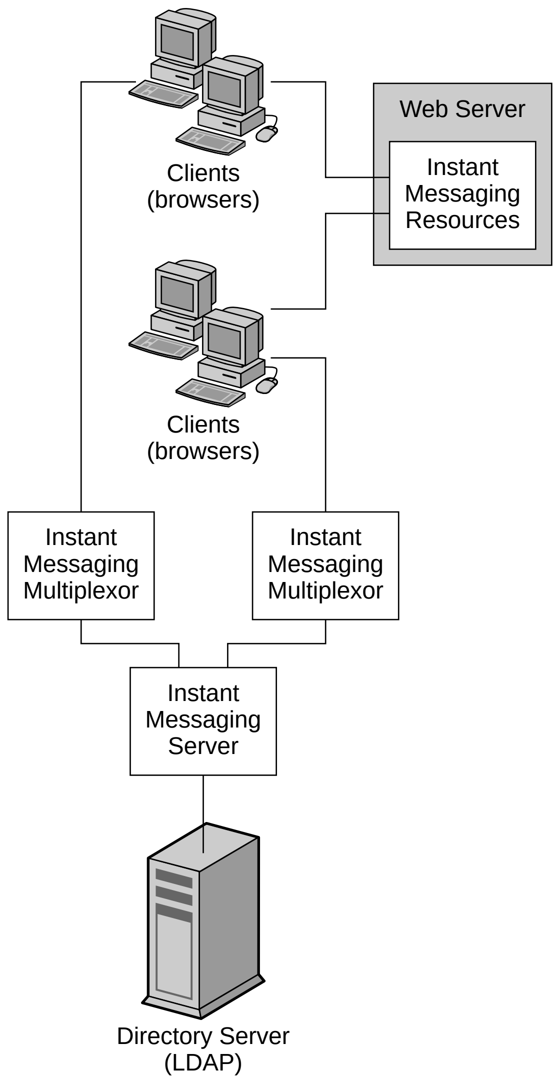
This chapter contains the following sections:

NOTE Currently, all deployment options are available on the Solaris platform. A subset of the deployment options is available on Linux and Windows operating systems.

Basic Instant Messaging Architecture

The following figure shows the basic Instant Messaging architecture.

Figure 23-1 Basic Instant Messaging Architecture



The basic Instant Messaging architecture provides such functionality as chat, news alerts, and conferences. To provide this basic functionality, you need to install the following components:

- Instant Messaging server and one or more Instant Messaging multiplexors
- Instant Messaging resources
- Web server such as Sun Java System Web Server
- LDAP server such as Sun Java System Directory Server

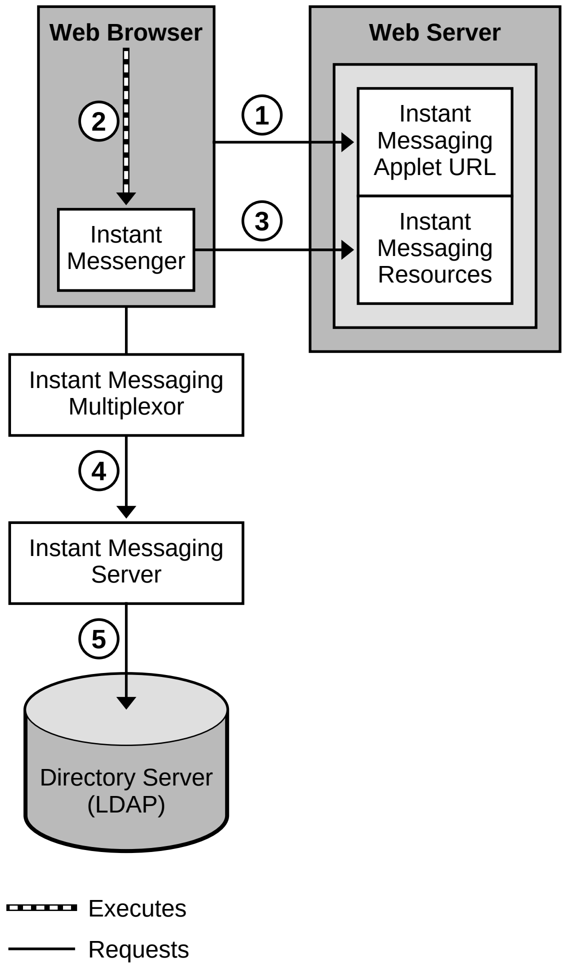
In this example:

- The LDAP server provides user entries for authentication and lookup.
- The clients download the Instant Messaging resources from either a web server or Sun Java System Application Server
- Clients always connect to the Instant Messaging server through an Instant Messaging multiplexor.

Authentication in a Basic Architecture

Figure 23-2 on page 314 shows the interaction of the software components in the authentication process of a basic architecture of Instant Messaging. The focus is on the flow of authentication requests. An explanation of the steps in this process follows the figure.

Figure 23-2 Flow of Authentication Requests in a Basic Instant Messaging Architecture



The authentication process in a basic architecture works as follows:

1. End user accesses the Instant Messenger applet URL from a browser and chooses a method to invoke the client.
2. The browser invokes Java Web Start or the Java plugin.
3. Java Web Start or the Java plugin downloads the necessary Instant Messenger resource files and starts Instant Messenger.

4. The login window appears and the end user enters the login name and password. The login data is sent to the Instant Messaging server through the multiplexor.

5. The Instant Messaging server communicates with the LDAP server to authenticate the end user and to request end-user information, such as contact lists or subscriptions.

When the end-user authentication is complete, the Instant Messaging main window appears, displaying the contact list for the end user. The end user can now start and participate in Instant Messaging sessions with the other end users.

Instant Messaging Email Notification (Calendar Alert) Architecture

You can deploy Instant Messaging to support email notification to offline users, as well as Instant Messaging based notification of calendar events to users.

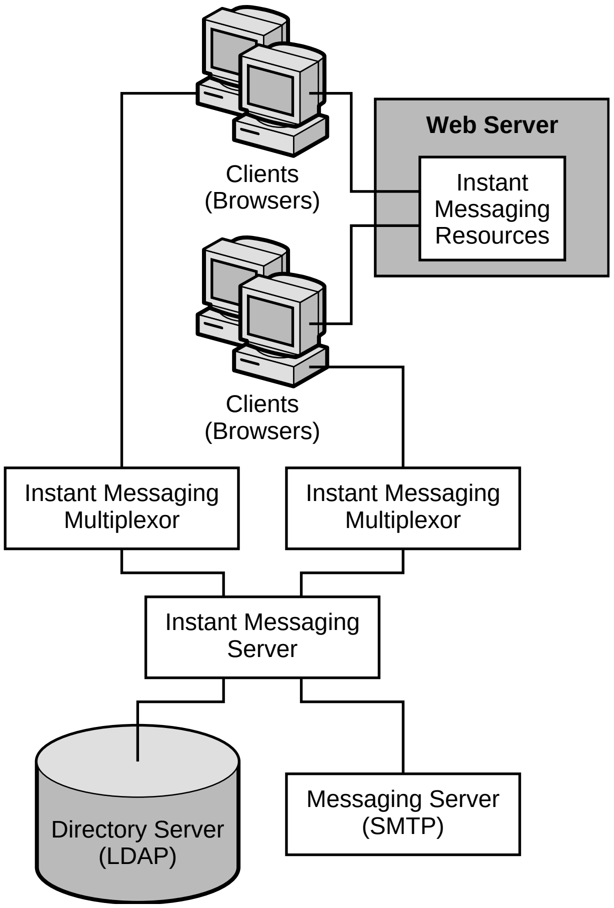
An Instant Messaging architecture that supports email notification and calendar alerts provides the same functionality as "Basic Instant Messaging Architecture" on page 312. To provide this functionality, you need to include the components listed in "Basic Instant Messaging Architecture" on page 312. To support email alerts, you also install an SMTP server such as Sun Java System Messaging Server. To support calendar alerts, you also install Sun Java System Calendar Server.

To enable email notification, you are prompted to identify the SMTP server to use with Instant Messaging during installation. If you do not have an SMTP server installed, you must install one before installing the Instant Messaging software. Figure 23-3 on page 316 shows Instant Messaging with email notification enabled on the network.

If you do not have Calendar Server installed, you must install it before installing the Instant Messaging software. Figure 23-4 on page 317 shows Instant Messaging with calendar notification enabled on the network.

Authentication flow in this architecture is the same as in a basic deployment. See "Authentication in a Basic Architecture" on page 313 for more information.

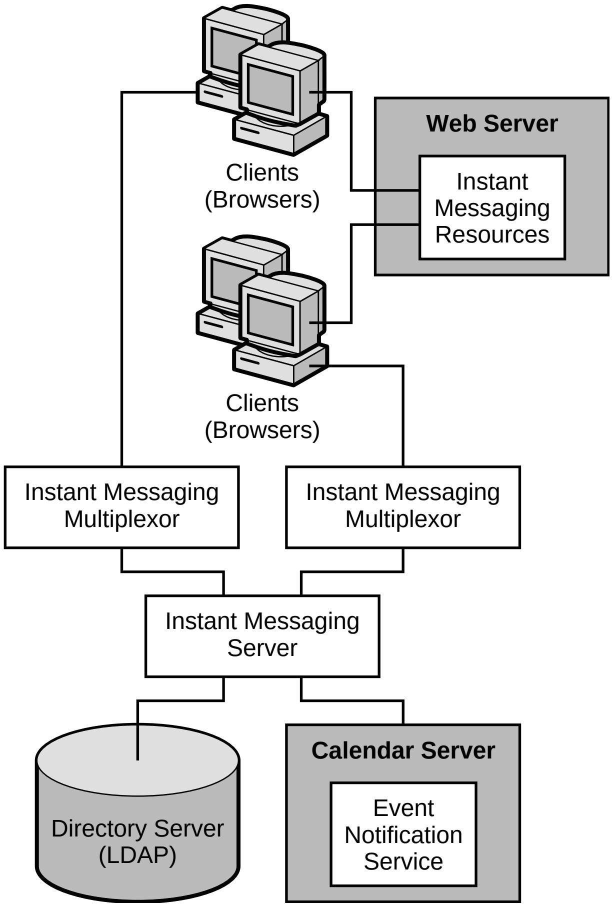
Figure 23-3 Instant Messaging Architecture with Email Notification



In this example:

- The LDAP server provides user entries for authentication and lookup.
- The Instant Messaging server forwards messages intended for offline users to the SMTP server. The SMTP server then sends the message as an email to the user's mailbox.
- The clients download the Instant Messaging resources from a web server (or application server).
- Clients always connect to the Instant Messaging server through an Instant Messaging multiplexor.

Figure 23-4 Instant Messaging Architecture with Calendar Alerts


Web Server
Instant Messaging Resources
Clients (Browsers)
Clients (Browsers)
Instant Messaging Multiplexor
Instant Messaging Multiplexor
Instant Messaging Server
Directory Server (LDAP)
Calendar Server
Event Notification Service

In this example:

- The LDAP server provides user entries for authentication and lookup.
- The Event Notification Server (ENS) sends notifications of calendar events to the Instant Messaging server which then forwards the notification on to the appropriate end user.
- The clients download the Instant Messaging resources from a web server (or application server).
- Clients always connect to the Instant Messaging server through an Instant Messaging multiplexor.

Instant Messaging Access Manager or SSO Architecture

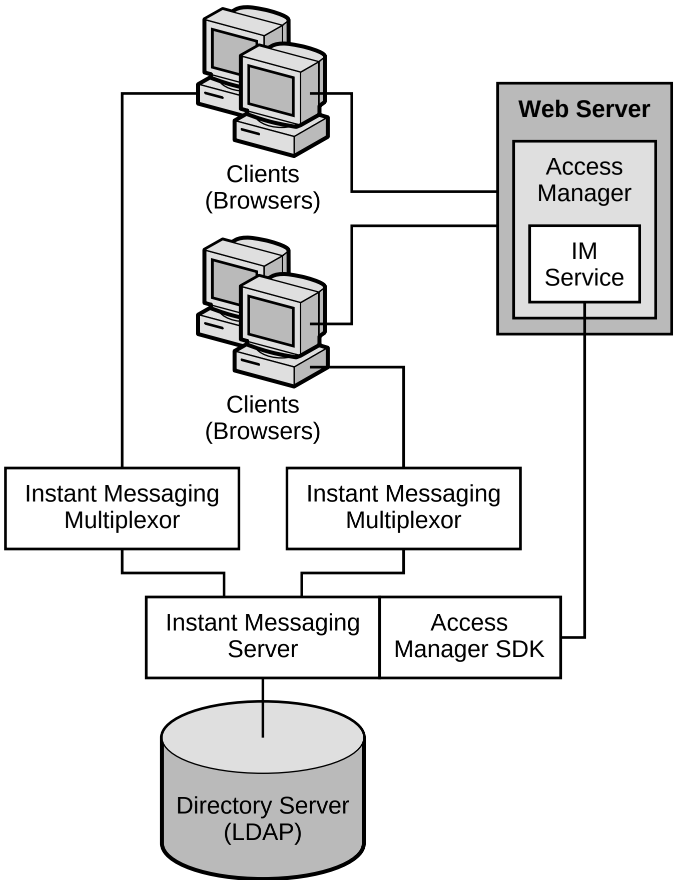
You can deploy Instant Messaging to use Access Manager policy features and single sign-on (SSO). An Instant Messaging architecture that uses Access Manager provides the same functionality as "Basic Instant Messaging Architecture" on page 312. To provide this functionality, you need to install the components listed in "Basic Instant Messaging Architecture" on page 312 and also install Access Manager. In addition, you need to install the Access Manager SDK on the Instant Messaging server host.

In this architecture, Instant Messaging uses the directory to search for users but not to authenticate or authorize them. Instead, Access Manager is responsible for authenticating and authorizing users.

If you are planning on using SSO with Access Manager, you must configure Access Manager and Instant Messaging to use the same web container.

Figure 23-5 on page 319 shows the Instant Messaging Access Manager architecture.

Figure 23-5 Instant Messaging Architecture With Access Manager-based Server Policy Management or Single Sign On



In this example:

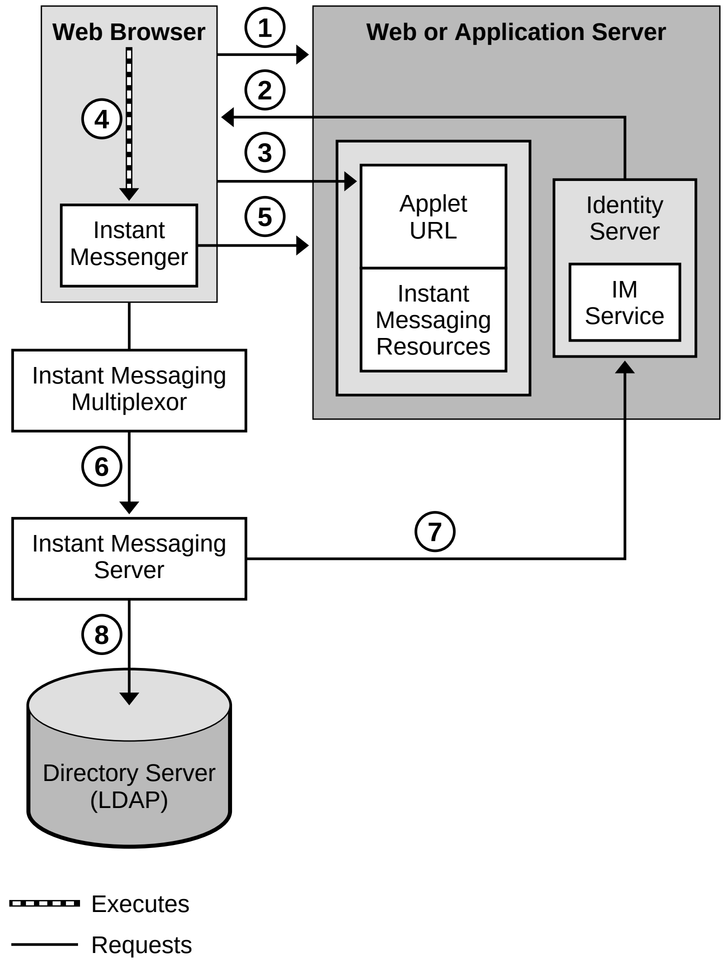
- The LDAP server provides user entries.
- The web server (can also be an application server, which includes a web server) downloads the Instant Messaging resources via a browser to the clients. The resources are basically the client.
- Clients always connect through a multiplexor.
- Instant Messaging related services provided by the Access Manager include the presence service and the Instant Messaging service.

- The web server can be used to access the Access Manager administration interface, which manages the identity-based services for the Instant Messaging deployment. The Web server for Access Manager can be the same as the one that serves the Instant Messaging resources. See the Access Manager documentation for more details.
- The Access Manager SDK provides the API used by the Instant Messaging server to interact with the Access Manager.

Authentication in an Access Manager Only Architecture

Figure 23-6 on page 321 illustrates the authentication process used by the Instant Messaging software in collaboration with Portal Server and Access Manager components in a single sign-on environment. As with Figure 23-2 on page 314, this figure focuses on the flow of authentication requests. An explanation of the steps in this process follows the figure.

Figure 23-6 Flow of Authentication Requests in an Access Manager Configuration



The authentication process of the Instant Messaging server in this deployment within a single sign-on environment works as follows:

1. The end user logs in to the Access Manager server by entering the URL in a web browser.

2. The Access Manager software authenticates the end user and returns a session token.

   The session token is what enables single sign-on to work. This token is provided as an applet parameter and is used throughout the authentication process. End users are not asked for their credentials again as long as the session token is present.

3. End user accesses the Instant Messenger applet URL from a browser and chooses a method to invoke the client.

4. The browser invokes Java Web Start or the Java plugin as appropriate.

5. Java Web Start or the Java plugin downloads the necessary Instant Messenger resource files and starts Instant Messenger.

6. Instant Messenger requests authentication to the Instant Messaging server using the session token.

7. The Instant Messaging server asks Access Manager to validate the session token. If the session is valid, Instant Messenger displays the end user's contact list and the end user can then use Instant Messenger services: chat, alerts, polls, etc.

8. The Instant Messaging server must query LDAP directly to get or set end-user information, such as contact lists or subscriptions.

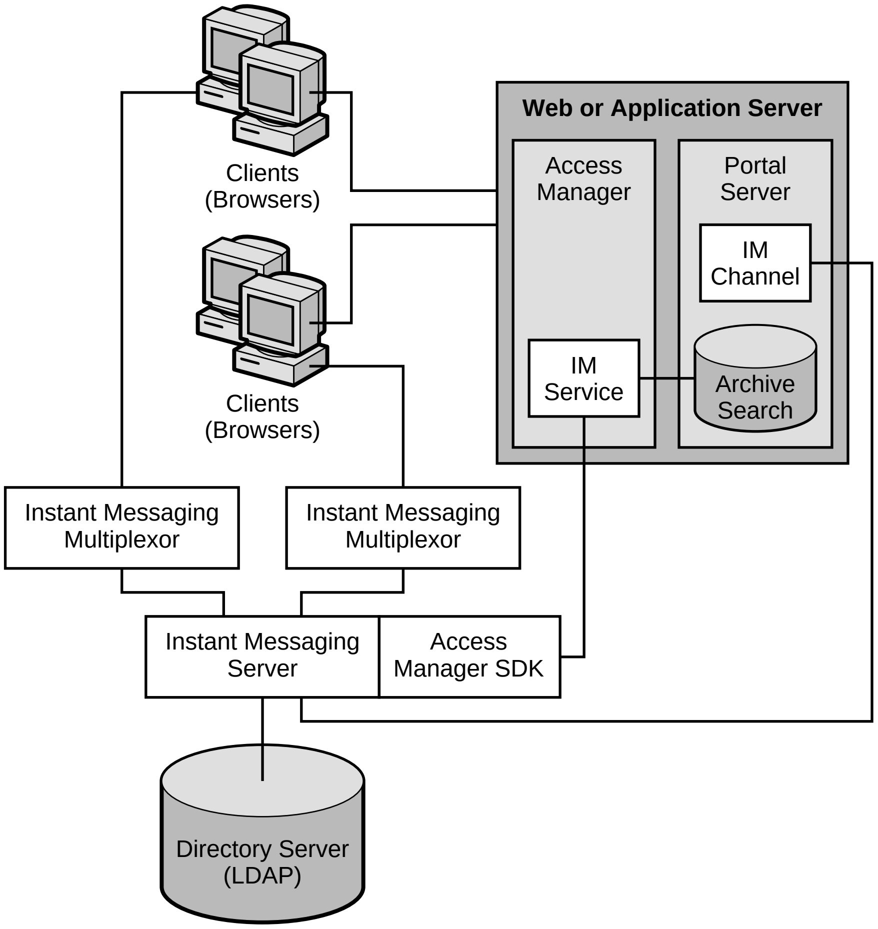
Instant Messaging Portal-based or Archiving Architecture

You can deploy Instant Messaging to support message archiving, and to run Instant Messaging in secure mode. An Instant Messaging architecture that provides for this functionality also provides the same functionality as "Basic Instant Messaging Architecture" on page 312. In addition, the Instant Messenger client is made available to end users through the Portal Server desktop. To provide this functionality, you need to install the components listed in "Basic Instant Messaging Architecture" on page 312 and also install Portal Server and Access Manager.

This architecture uses the directory and Web servers accessed by the Access Manager. You do not need to install additional instances of those servers. In addition, because this architecture requires Access Manager, all the features described in "Instant Messaging Access Manager or SSO Architecture" on page 318 are also available.

The following figure shows the Instant Messaging Portal-based architecture.

Figure 23-7 Instant Messaging Architecture With Portal-based Secure Mode or Archiving



In this example:

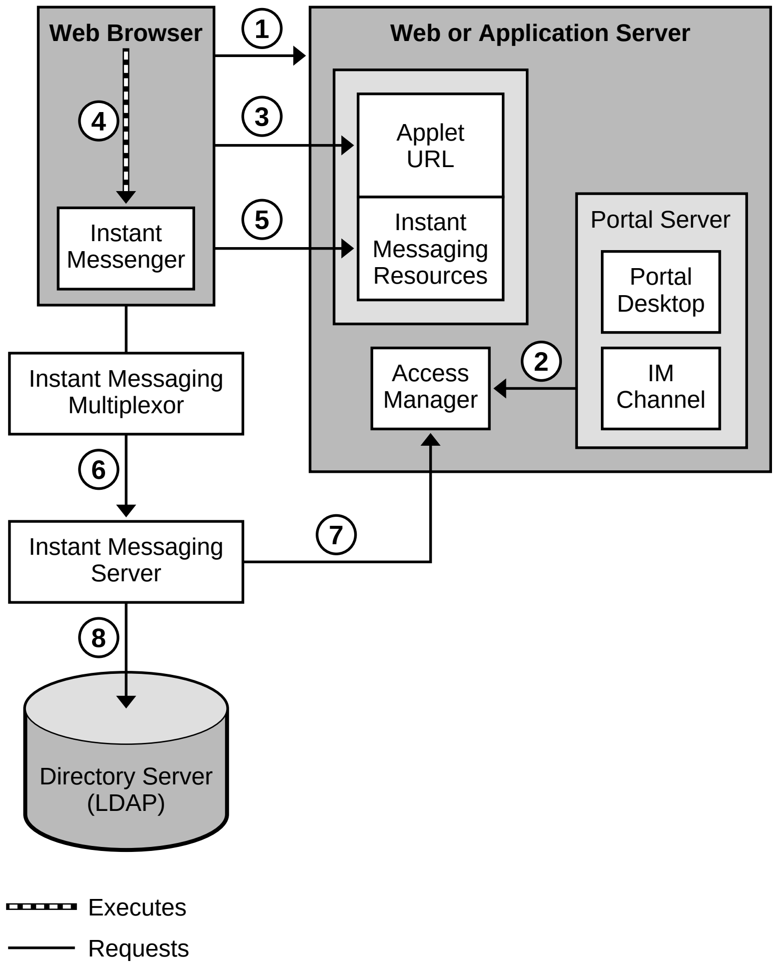
- The LDAP server provides user entries.
- The web server (can also be an application server, which includes a web server) downloads the Instant Messaging resources via a browser to the clients. The resources are basically the client.
- Clients are always connect through an Instant Messaging multiplexor.

- Instant Messaging related services provided by Access Manager include the presence service and the Instant Messaging service.
- The web server can be used to access the Access Manager administration interface, which manages the identity-based services for the Instant Messaging deployment. The web server for both Access Manager and Portal Server can be the same as the one that serves the Instant Messaging resources. See the Sun Java System Access Manager and Sun Java System Portal Server documentation for more details.
- The Access Manager SDK provides the API used by the Instant Messaging server to interact with the Access Manager.
- The Instant Messaging channel is supported by Portal Server and enables users to access Instant Messenger from the Portal Desktop.
- Portal Server provides archive functionality that allows you to save instant messages sent through the deployment.

Authentication in a Portal Server Architecture

Figure 23-8 on page 325 illustrates authentication process used by the Instant Messaging software in collaboration with Portal Server and Access Manager components in a single sign-on environment. As with Figure 23-2 on page 314, this figure focuses on the flow of authentication requests. An explanation of the steps in this process follows the figure.

Figure 23-8 Flow of Authentication Requests in a Portal Server and Access Manager Configuration



The authentication process of the Instant Messaging server in this deployment within a single sign-on environment works as follows:

1. The end user logs in to the Portal Server by entering the URL in a web browser.
2. The Access Manager software authenticates the end user and returns a session token and the Portal Server downloads the Desktop for the end user. The Portal Server Desktop is displayed in the end user's browser. See Step 6 for an explanation of the session token.
3. The end user clicks the Instant Messenger URL link from the Instant Messaging channel on the Desktop.

4. The browser invokes Java Web Start or the Java plugin.
5. Java Web Start or the Java plugin downloads the necessary Instant Messenger resource files and starts the Instant Messenger.
6. Instant Messenger requests authentication to the Instant Messaging server using the session token.

   The session token is what enables single sign-on to work. This token is provided as an applet parameter and is used throughout the authentication process. End users are not asked for their credentials again as long as the session token is present.
7. The Instant Messaging server asks Access Manager to validate the session token. If the session is valid, Instant Messenger displays the end user's contact list and the end user can then use Instant Messenger services: chat, alerts, polls, and so forth.
8. The Instant Messaging server must query LDAP directly to get or set end-user information, such as contact lists or subscriptions.

Instant Messaging With All Features Enabled

You can deploy Instant Messaging and enable all the features listed in this section by installing:

- The following components before you install Instant Messaging:
  - Directory Server (during Access Manager installation)
  - Web Server (during Access Manager installation)
  - Access Manager
  - Portal Server
  - Calendar Server
  - Messaging Server
- Instant Messaging resources on the Web Server host
- Access Manager SDK on the Instant Messaging server host

You also need to configure the Access Manager Instant Messaging Service on the Access Manager host.

Instant Messaging Physical Deployment Examples

This section explains variations to the deployment scenario described in "Basic Instant Messaging Architecture" on page 312. For example, you can install the various required servers and components in the following physical configurations:

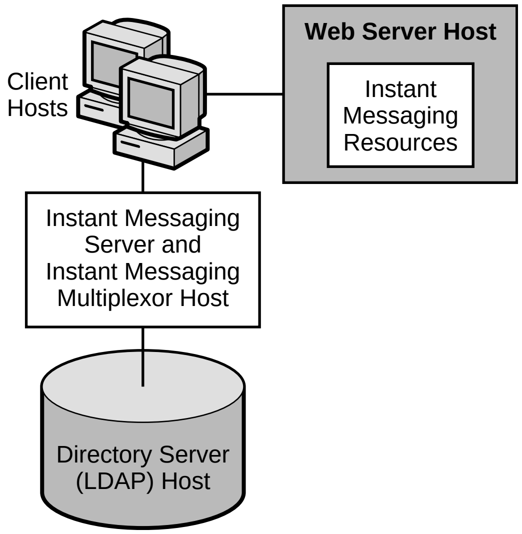
- Instant Messaging Physical Deployment Example: Web Server on Separate Host
- Instant Messaging Physical Deployment Example: Multiplexors on Separate Hosts
- Instant Messaging Physical Deployment Example: Multiple Instant Messaging Hosts
- Any combination or all of the above

These variations can be applied in any of the architectures described in this chapter. You might choose to include them based on your deployment requirements.

Instant Messaging Physical Deployment Example: Web Server on Separate Host

Figure 23-9 on page 328 shows a configuration consisting of the Instant Messaging server and multiplexor installed on the same host. The web server is installed on a separate host. The Instant Messaging resources are also present on the web server host. Use this configuration when there is an existing instance of a web server and an LDAP server, and you do not want to install other applications on these hosts.

Figure 23-9 Separate Web Server and Instant Messaging Hosts


Client Hosts
Web Server Host
Instant Messaging Resources
Instant Messaging Server and Instant Messaging Multiplexor Host
Directory Server (LDAP) Host

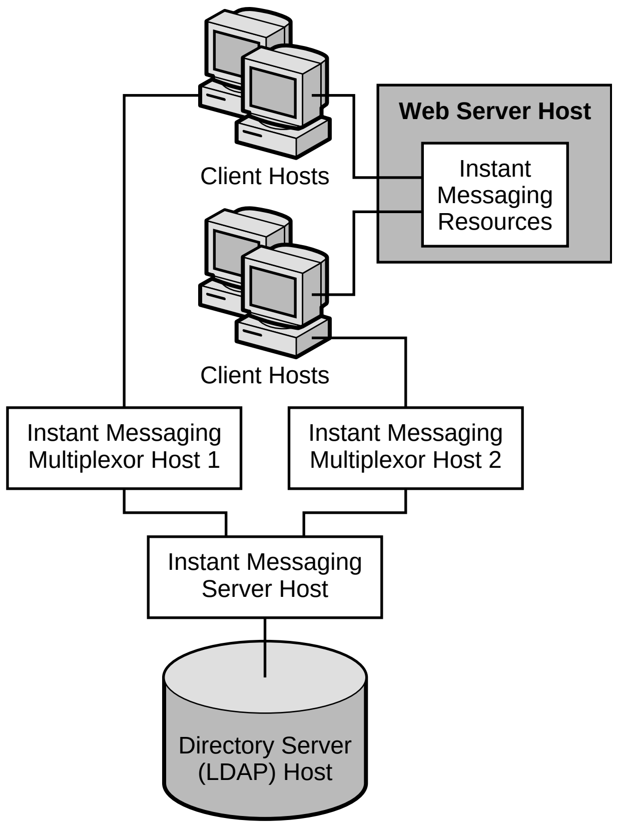
Instant Messaging Physical Deployment Example: Multiplexors on Separate Hosts

Figure 23-10 on page 329 shows a configuration consisting of two multiplexors installed on two separate hosts.The Instant Messaging server is installed on a different host. This configuration enables you to place a multiplexor outside your company's firewall. Installing multiplexors on multiple hosts distributes the load of the Instant Messaging server across multiple systems.

NOTE The multiplexor can be resource-intensive, so putting it on a separate host can improve the overall performance of the system.

Windows supports only one multiplexor instance per host.

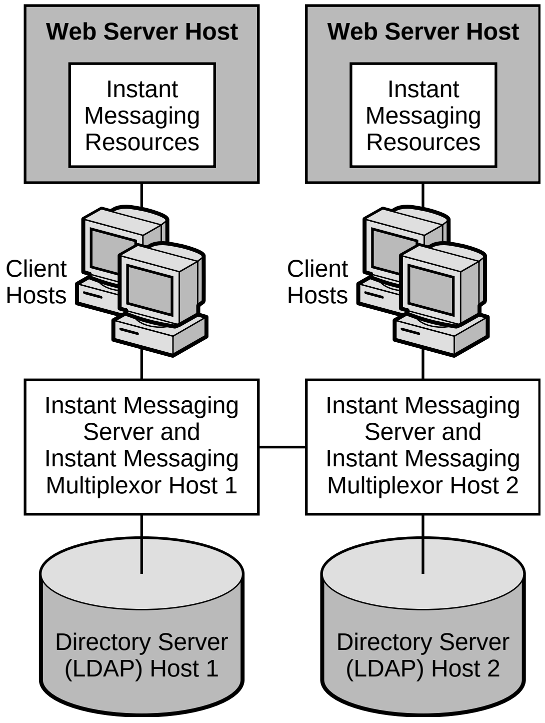
Figure 23-10 Instant Messaging Multiplexors Installed on Separate Hosts


Client Hosts
Web Server Host
Instant Messaging Resources
Client Hosts
Instant Messaging Multiplexor Host 1
Instant Messaging Multiplexor Host 2
Instant Messaging Server Host
Directory Server (LDAP) Host

Instant Messaging Physical Deployment Example: Multiple Instant Messaging Hosts

Figure 23-11 on page 330 shows a configuration consisting of two Instant Messaging servers. This configuration is used when the site contains multiple administrative domains. The server configuration on each Instant Messaging server host has to be set up so that end users on one Instant Messaging server can communicate with end users on other Instant Messaging servers.

Figure 23-11 Multiple Instant Messaging Server Hosts


Web Server Host
Instant Messaging Resources
Web Server Host
Instant Messaging Resources
Client Hosts
Client Hosts
Instant Messaging Server and Instant Messaging Multiplexor Host 1
Instant Messaging Server and Instant Messaging Multiplexor Host 2
Directory Server (LDAP) Host 1
Directory Server (LDAP) Host 2

Chapter 24

Understanding Instant Messaging Pre-Installation Considerations

This chapter describes considerations you need to think about before installing Instant Messaging. See the *Sun Java Enterprise System 2005Q1 Installation Guide* for instructions on running the Java Enterprise System installer.

This chapter contains the following sections:

- Installing Instant Messaging Overview
- Instant Messaging Worksheets

Installing Instant Messaging Overview

To install Instant Messaging on a Solaris system, you use the Java Enterprise System installer. On Linux and Windows systems, you use the setup program included in the Linux and Windows Media Kit CD. Optionally, you can download the software from the following site:

`http://www.sun.com/software/download/`

The Java Enterprise System and Instant Messaging documentation provide procedures and tools for completing and upgrading your installation, for configuring your servers, setting up clients, and so on. For more information on these additional installation and configuration steps, see the following:

Sun Java Enterprise System 2005Q1 Installation Guide
`http://docs.sun.com/doc/819-0056`

Sun Java Enterprise System 2005Q1 Upgrade and Migration Guide
`http://docs.sun.com/doc/819-0062`

Sun Java System Instant Messaging Administration Guide
`http://docs.sun.com/doc/819-0430`

Before you begin an installation, see the Instant Messaging Release Notes for hardware and software requirements and supported versions:

`http://docs.sun.com/doc/819-0428`

Prior to installing Instant Messaging, you need to install a directory server, web server, and optionally a messaging server. On Solaris systems, you might also need to install Access Manager and Portal Server if you intend to use functionality provided by those servers with Instant Messaging. Interoperating with other servers is described in "Components Related to Instant Messaging" on page 283. In addition, Chapter 23, "Developing an Instant Messaging Architecture" provides some architectures that you can follow to enable various Instant Messaging features.

Instant Messaging Worksheets

During installation and upgrade, you will be prompted for basic configuration information. You should gather this information before you begin. You will be prompted for some or all of the information depending on the components you decide to install.

Print out Table 24-1 and write the values for your deployment in the space provided. You can reuse this installation worksheet for multiple installations, uninstallation, or for upgrades. This table contains passwords and other sensitive information, so you should store this information in a safe place.

Table 24-1 Communications Services Installation Parameters

Parameter	Description	Your Answers
Installation Directory	*instant-messaging-install-dir* or *installation directory*. Directory in which Instant Messaging is installed. Defaults: Solaris system: `/opt/SUNWiim` Linux system: `/opt/sun/im` Windows system: `C:\Program Files\Sun\Instant Messaging`	

Table 24-1 Communications Services Installation Parameters *(Continued)*

Parameter	Description	Your Answers
Instant Messaging Server Host and Domain Name	Host name on which Instant Messaging is installed and the domain name associated with the host. For example: Host Name: `instantmessaging.siroe.com` Domain Name: `siroe.com`	
Instant Messaging Server Port Number	The port number on which the Instant Messaging server listens for incoming requests other than those sent by Instant Messenger clients. Default: 49999	
Multiplexor Port Number (Multiplexor Configuration Only)	The port number on which the Instant Messaging server listens for incoming requests from Instant Messenger clients. Default: 49909	
Disable Server	Select this option if the instance you installed will act as a multiplexor and not a server. If you select this option, you must provide a value for Remote Instant Messaging Server Host Name (Multiplexor Configuration Only).	
Remote Instant Messaging Server Host Name (Multiplexor Configuration Only)	The host name of the Instant Messaging server for which this multiplexor routes messages. Do not enter a value for this parameter if the installed instance you are configuring is an Instant Messaging server and not a multiplexor. Dependencies: The Disable Server parameter must be selected, that is, server functionality is disabled.	
Assign Instant Messaging Services to existing users (Optional)	If selected, this option enables Instant Messaging for existing Access Manager users. Dependencies: Portal Server and Access Manager.	

Table 24-1 Communications Services Installation Parameters *(Continued)*

Parameter	Description	Your Answers
Secure Mode (Optional)	When selected, enables integration with Sun Java System Portal Server Secure Remote Access. Secure Remote Access provides secure access to remote users in an intranet. Users can access Secure Remote Access by logging in to the web-based Portal Server Desktop through the portal gateway. Dependencies: Requires Portal Server and Access Manager. You can run Instant Messaging in secure mode only if Secure Remote Access is configured. See the *Sun Java System Instant Messaging Administration Guide* and *Sun Java System Portal Server Secure Remote Access Administration Guide* for instructions. If you enable this feature, you must provide values for the following parameters: • Netlet Instant Messaging Port Number (Optional) • Messenger Secure Download Port (Optional)	
Netlet Instant Messaging Port Number (Optional)	If you enabled Secure Mode (Optional), this is the port number on which Netlet listens for incoming requests. Default: 49917 Dependencies: Secure Mode (Optional) enabled, Portal Server, and Access Manager.	
Messenger Secure Download Port (Optional)	If you enabled Secure Mode (Optional), this is the port number from which Instant Messenger resources will be downloaded through Netlet. Default: 49916 Dependencies: Secure Mode (Optional) enabled, Portal Server, and Access Manager.	
Enable Instant Messaging Archive (Optional)	If selected, enables Portal Server search-based archiving for Communications Services. Dependencies: Portal Server and Access Manager.	
LDAP Host Name	The host name of the LDAP server that contains user and group information for Communications Services. For example, `directory.siroe.com`. Dependencies: LDAP server such as Directory Server.	

Table 24-1 Communications Services Installation Parameters *(Continued)*

Parameter	Description	Your Answers
LDAP Port Number	The port number on which the directory server listens for incoming requests. For example, `389`. Dependencies: LDAP server such as Directory Server.	
Base DN	The base distinguished name in the directory tree that contains user and group information for Instant Messaging. For example, `o=siroe.com`. Dependencies: LDAP server such as Directory Server.	
Bind DN	During installation, you must use the Directory Manager Bind DN and password. The information is used to update the directory schema with the Instant Messaging and presence service templates and attributes only. This requires Directory Manager access. The Directory Manager Bind DN and password are not saved or used beyond installation and initial configuration. For server configuration, Instant Messaging uses this Bind DN to search users and groups in the directory. Leave this blank if the directory can be searched anonymously. Dependencies: LDAP server such as Directory Server.	
Bind Password	The Bind DN password.	
SMTP Server Host Name (Optional)	The host name of the SMTP server used to send email notification of messages to offline users. For example, `mail.siroe.com`. If the SMTP server does not use port 25, specify the port along with the host name. For example, if the SMTP server uses port `1025`: `mail.siroe.com:1025` Dependencies: SMTP server such as Messaging Server.	
Database, Logs, and Runtime File Pathname	The location where the runtime files, database, and logs are stored. Defaults: Solaris system: `/var/opt/SUNWiim/default` Linux system: `/var/opt/sun/im` Windows system: `C:\Program Files\Sun\Instant Messaging`	

Table 24-1 Communications Services Installation Parameters *(Continued)*

Parameter	Description	Your Answers
Resources and Help Files Pathname	*instant-messaging-resource-directory* or *resource directory* The directory in which the resource and online help files are installed. Defaults: Solaris system: `/opt/SUNWiim/html` Linux system: `/opt/sun/im/html` Windows system: `C:\Program Files\Sun\Instant Messaging\html`	

Code Base	The URL from which Instant Messenger downloads resources.
	
	You install the resources into the web server's doc root. For example, assume that the web server `www.example.com` listens on port `89`, the doc root for this web server is `/opt/web/`, and you choose to install the messenger resources in `/opt/web/im`, then the messenger resources codebase is as follows:
	
	`http://www.example.com:89/im/`
	
	If you do not provide the correct `codebase` during installation, you need to update the messenger launch pages *codebase*`/`*lang*`/im[ssl].html` and *codebase*`/`*lang*`/im[ssl].jnlp` with the correct URL.
	
	On UNIX, it is possible to install the resources in a directory and use a symbolic link to make the resources visible to the web server.
	
	For instance, if in the above example you installed the resources in `/opt/SUNWiim/html`, the messenger resources can be made visible to the web server by creating the following symbolic link.
	
	`ln -s /opt/SUNWiim/html /opt/web/im`
	
	See the *Sun Java System Instant Messaging Administration Guide*, and your web server documentation for more information.	

Part V

Deploying Communications Express

Chapter 25

Introduction to Communications Express Software

Communications Express provides an integrated web-based communications and collaboration client. Communications Express is a common part of Messaging Server and Calendar Server, providing end users with a web interface to their calendar information and mail, as well as an address book.

Communications Express consists of three client modules: Calendar, Address Book, and Mail.

This chapter contains the following sections:

- Communications Express Overview
- Communications Express Features
- Communications Express High-Level Architecture

Communications Express Overview

Communications Express depends upon the following Sun Java System component products:

- Directory Server
- Access Manager (if you are using Sun Java System LDAP Schema Version 2)
- Calendar Server
- Messaging Server
- Web Server or Application Server (for the web container)

You install Communications Express as a front-end server (in a multi-tier environment). You must install the complete set of Messaging Server packages on the same host that Communications Express is running on. Also, both Communications Express and Messenger Express must run on the same IP address. The Messaging Server packages can then be configured to run as Messenger Express or as MEM, which connects to a back-end store running Messenger Express.

In addition, you can configure Communications Express to have the Address Book on the front-end machine store its data either in the LDAP directory infrastructure or on an LDAP server other than the Communications Express machine. See the *Sun Java Communications Express Administration Guide* for more information:

`http://docs.sun.com/doc/819-0115`

Communications Express communicates with Calendar Server through the Calendar Server HTTP service, the `mshttpd` daemons for Messaging Server, and the LDAP service for address book. The `cshttpd` daemon can be local or remote, the `mshttpd` daemon can be either the local Webmail server or MEM, and the LDAP service can be either local or remote.

When using a load balancer or port director type device, make sure to utilize "sticky" (persistent) connections such that users are continually routed to the same front-end server for the duration of their session.

Communications Express Features

- Communications Express has an integrated user interface for calendar, mail, and address book and enables the access of one client module from another without re-authenticating user credentials.
- Communication between mail and calendar is established using Access Manager or Messaging Server single sign-on mechanism.
- Both calendar and mail applications share the same address book.
- All modules share the common user preferences specified in the Options tab of Communications Express.
- The Address Book Store provides horizontal scalability. See the *Sun Java Communications Express Administration Guide* for more information:

  `http://docs.sun.com/doc/819-0115`
- Communications Express supports virtual domains.

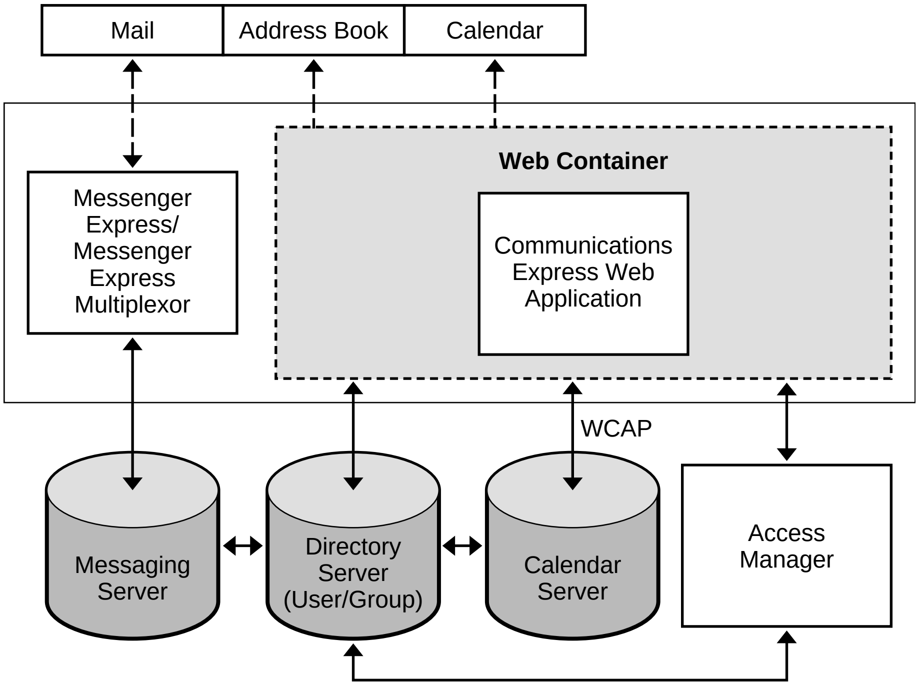
Communications Express High-Level Architecture

The Calendar and Address Book client modules are deployed as a single web application in a web container, which can be either Sun Java Systems Web Server or Sun Java Systems Application Server. The mail module is rendered by Messenger Express. Messenger Express is the standalone web-based mail application that uses the HTTP service of the Messaging Server.

Messenger Express or MEM should be deployed on the same system where Communications Express is deployed.

The following figure shows the Communications Express software architecture.

Figure 25-1 High Level Communications Express Software Architecture


Mail
Address Book
Calendar
Web Container
Messenger Express/ Messenger Express Multiplexor
Communications Express Web Application
WCAP
Messaging Server
Directory Server (User/Group)
Calendar Server
Access Manager

Communications Express consists of the following modules:

- **Mail.** The Mail component uses the JavaScript language that is read and interpreted by the client. The JavaScript files are located on the server and downloaded to the client. The client extracts data from the JavaScript code to customize Communications Express functions. All modifications and customizations are done on the server.
- **Calendar.** The presentation layer of the Calendar module is based on JavaServer Pages™. These JavaServer Pages pages can be customized to suit the requirements of the client. The data layer accesses a Java API for Calendar (JCAPI) to enable exchange of data with Calendar Server over HTTP-based protocol.
- **Address Book.** The Address Book component uses XML / XSL files that contain XSL tags, static HTML and `.js` scripts. The XSL and JavaScript code are used to display dynamic data. These XSL files can be edited for customizing the Address Book component.

Chapter 26

Developing a Communications Express Architecture

This chapter contains Communications Express basic deployment architectures. Depending on the features you want to implement in your deployment, you will need to install different sets of hosts and other networking infrastructure.

This chapter contains the following sections:

- Basic Communications Express Architecture
- Communications Express on Remote Host Architecture

Basic Communications Express Architecture

This basic Communications Express architecture provides Calendar, Address Book, and Mail modules in a web container on a single host. Messenger Express is the standalone web interface mail application that uses the HTTP service of the Messaging Server. Messenger Express is deployed on the same system as the Calendar and Address Book modules.

To provide this basic functionality, you need to install the following components:

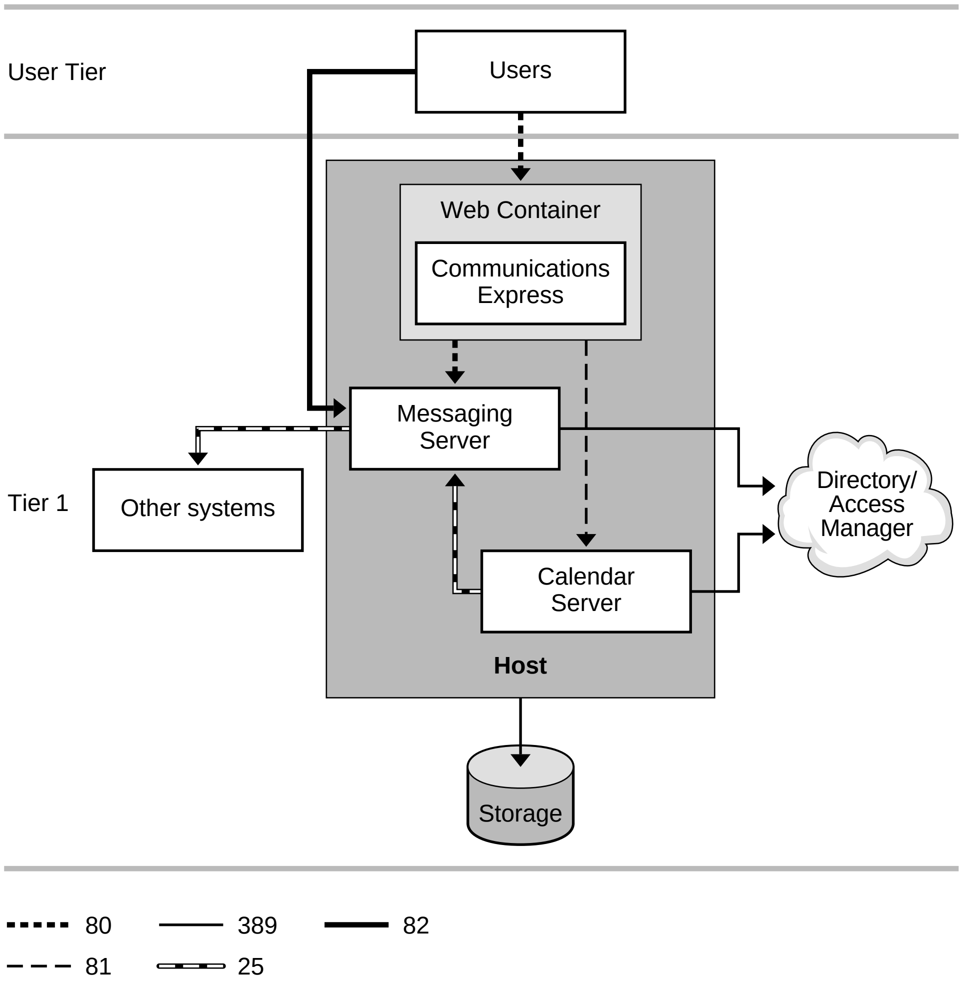
- Directory Server
- Access Manager (If you are using Sun Java System LDAP Schema Version 2)
- Calendar Server
- Messaging Server
- Web Server or Application Server (for the web container)

In this example:

- You install the complete set of Messaging Server packages on the host that Communications Express is running on.
- The AddressBook server of Communications Express is configured to store its data in the LDAP directory infrastructure.
- SSL has not been configured.

The following figure shows the basic Communications Express architecture.

Figure 26-1 Basic Communications Express Architecture



The following table explains the protocols and port numbers used by this architecture.

Table 26-1 Protocols And Ports Used by Basic Communications Express Deployment Architecture

Protocol	**Port**	**Used By**
SMTP	25	Messaging Server MTA component to communicate with other systems, and Calendar Server (csenpd) components for email notifications
HTTP	80	Internet users to communicate with Communications Express front-end, and Communications Express to communicate with Messaging Server
HTTP	81	Calendar Express on Communications Express to communicate with Calendar Server
MSHTTP	82	Internet users to communicate with Messenger Express
LDAP	389	Messaging Server and Calendar Server to communicate with LDAP directory

Communications Express on Remote Host Architecture

Figure 26-2 on page 347 shows a Communications Express architecture for both intranet and Internet users. The intranet users log on to the Communications Express back-end host. The Internet users log on to the Communications Express front-end host in the DMZ, which then communicates with the back-end host. Single sign-on is enabled on the back-end host.

You install the front-end host with the following components:

- Communications Express
- Web container
- Messaging Express Multiplexor
- Access Manager SDK

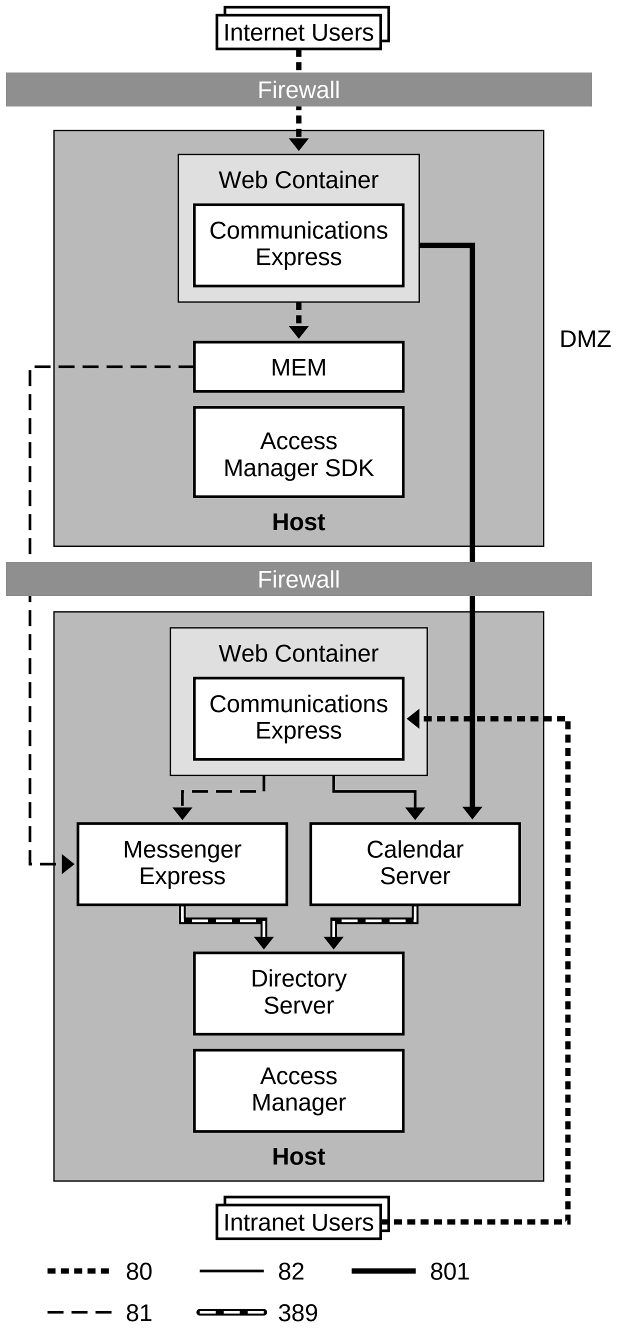
You install the back-end with the following components:

- Communications Express
- Web container

- Messaging Server (Messenger Express)
- Calendar Server
- Directory Server
- Access Manager

Figure 26-2 on page 347 shows the Communications Express on remote host architecture.

Figure 26-2 Communications Express on Remote Host Architecture



The following table explains the protocols and port numbers used by this architecture.

Table 26-2 Protocols And Ports Used by Communications Express Remote Host Deployment Example

Protocol	Port	Used By
HTTP	80	Internet users to communicate with the Communications Express front-end host in the DMZ
HTTP	81	Messaging Express Multiplexor (MEM) on the Communications Express front-end host in the DMZ to communicate with Messenger Express on the back-end host behind the DMZ
HTTP	82	Communications Express on the back-end host to communicate with Calendar Server, also on the back-end host
LDAP	389	Messaging Server and Calendar Server to communicate with LDAP directory
HTTP	8081	Communications Express on the front-end host to communicate with Calendar Server on the back-end host

Chapter 27

Understanding Communications Express Pre-Installation Considerations

This chapter describes considerations you need to think about before installing Communications Express.

This chapter contains the following sections:

- Communications Express Installation Considerations
- Requirements for Using S/MIME with Communications Express Mail

Communications Express Installation Considerations

Before installing Communications Express, consider the following planning aspects:

- Delegated Administrator requires that you install Access Manager and the web container (either Web Server or Application Server) on the same host.
- You can deploy Communications Express and Access Manager in both SSL and non-SSL modes, either on the same or a different web container.
- Due to a JavaScript security dependency, you must install Communications Express and Messenger Express on the same host, or Communications Express and Messaging Express Multiplexor on same host (in a multi-tiered environment).

- You can plan for a distributed deployment in which Directory Server, Messaging Server, Calendar Server, and Access Manager are installed on separate hosts.
- If you are using Calendar Server hosted domains, you enable Communications Express support for hosted domains during the configuration phase.
- You can configure Communications Express for SSL or non-SSL. If you configure SSL, you can choose between having Communications Express clients use SSL only for authentication, or to use SSL for the entire session.

Requirements for Using S/MIME with Communications Express Mail

Communications Express Mail now includes the security advantages of the Secure/Multipurpose Internet Mail Extension (S/MIME). Communications Express Mail users who are set up to use S/MIME can exchange signed or encrypted messages with other Communications Express Mail users, and with users of the Microsoft Outlook mail system or other mail clients that support S/MIME.

General Requirements for S/MIME

The signature and encryption features of S/MIME are available to a Communications Express Mail user only after:

- A private and public key pair are issued with a certificate in standard X.509 format. The certificate assures other mail users that the keys really belong to the person who uses them. Keys and their certificate are issued from within your organization or purchased from a third-party vendor. Regardless of how the keys and certificate are issued, the issuing organization is referred to as a certificate authority (CA).
- The private-public key pair, with its certificate, are properly stored electronically in a local key store or distributed to end users on common access cards (CACs), referred to as smart cards.
- All public keys and certificates are stored to an LDAP directory, accessible by Directory Server. This is referred to as publishing the public keys to make them available to other mail users who are creating S/MIME messages.

- Card reading devices are properly installed on the client machines when private-public key pairs and their certificates are stored on smart cards.
- All the necessary platform software is installed on the client machines where Communications Express Mail is accessed.
- All the necessary Sun Microsystems software is installed and configured for S/MIME.
- The Communications Express Mail user is set up to use the Sun Microsystems mail system. This includes giving the user permission to use the S/MIME features.

Concepts You Should Know Before Deploying S/MIME

Before you deploy your mail system for S/MIME, be sure you are familiar with these concepts:

- Basic administrative procedures of your platform
- Structure and use of an LDAP directory
- Addition or modification of entries in an LDAP directory
- Configuration process for Sun Java System Directory Server
- Concepts and purpose of the following:
    - Secure Socket Layer (SSL) for a secured communications line
    - Digitally signed email messages
    - Encrypted email messages
    - Local key store of a browser
    - Smart cards and the software and hardware to use them
    - Private-public key pairs and their certificates
    - Certificate authorities (CA)
    - Verifying keys and their certificates
    - Certificate revocation list (CRL)

Where to Go for More Communications Express Information

To install and configure Communications Express, see the instructions in the *Sun Java Systems Communications Express Administration Guide*:

http://docs.sun.com/doc/819-0115

To administer S/MIME, see the Administering S/MIME for Communications Express Mail chapter in the *Sun Java Systems Messaging Server Administration Guide*:

http://docs.sun.com/doc/819-0105

Part VI

Deployment Examples

Chapter 28

Communications Services Deployment Examples

This chapter contains Communications Services deployment examples. Depending on the features you want to implement in your deployment, you will need to install different sets of hosts and other networking infrastructure.

This chapter contains the following sections:

- Communications Services Single-tiered Logical Deployment Example for One Host
- Communications Services Two-tiered Logical Deployment Example for Multiple Hosts

NOTE When making architectural decisions, ranging from single host deployment, to multi-tiered deployments, you should always plan for service definition across multiple tiers. Thus, use logical service names to install, even on a single host deployment. Logical service names position deployments for easier expansion. See "Using Logical Service Names" on page 95 for more information.

Communications Services Single-tiered Logical Deployment Example for One Host

As its name implies, this example installs and configures components onto a single server. Consult with your Sun Client Services representative to determine the server type and configuration that best suits your needs.

In general, the single-tiered one host architecture is best suited for enterprises that are:

- Composed of less than 1,000 users
- Not geographically distributed
- Served by few administrators
- Seeking an entry-level configuration

The trade-offs associated with a single-host configuration include:

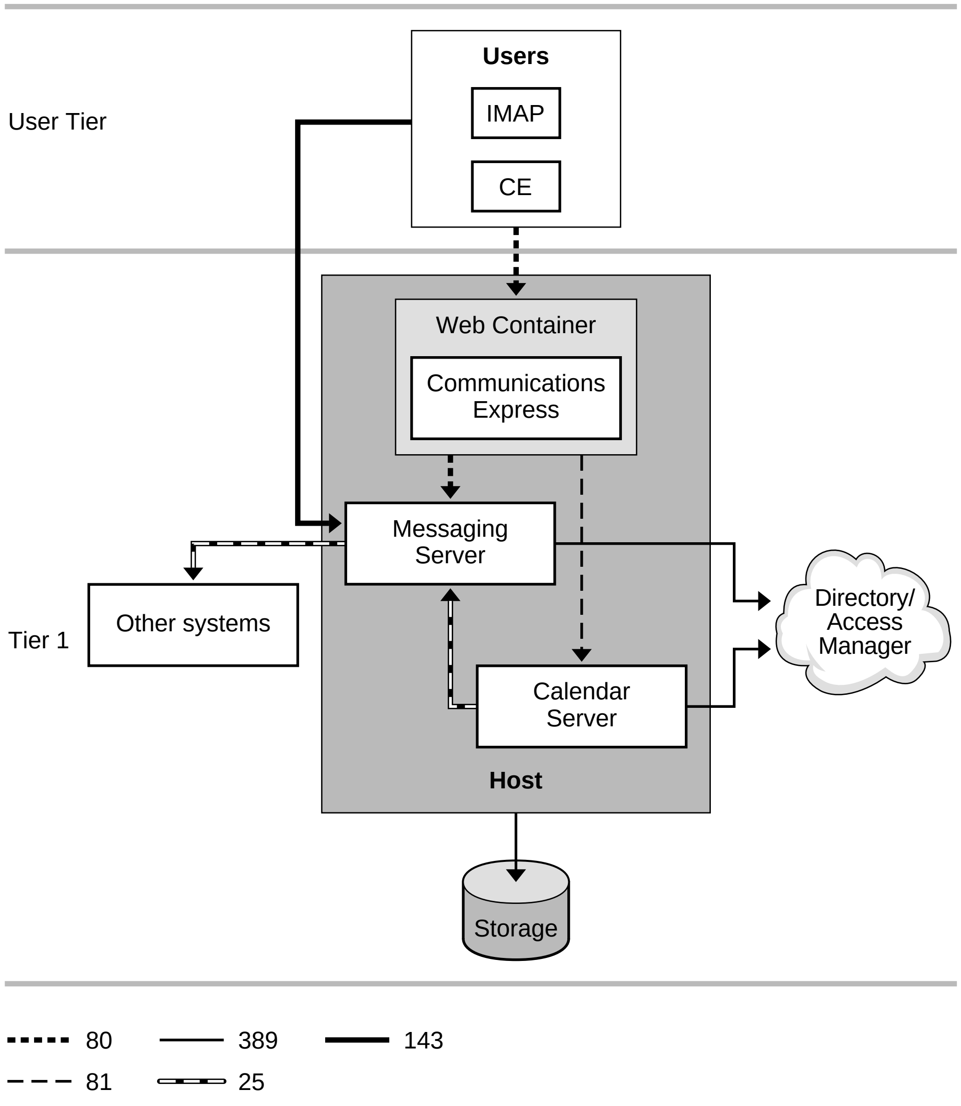
- No high availability to the infrastructure to ensure service reliability (unless the server itself provides automatic system reconfiguration)
- No ability to counteract denial of service attacks

Figure 28-1 on page 357 shows a single-host deployment example. The following Communications Services components are installed on the same host:

- Messaging Server (MTA, Message Store, and Messenger Express)
- Calendar Server (Administration service, HTTP service, and Backup service)
- Communications Express
- Web Server

In this example, the directory service resides on a different host than Communications Services. Directory Server and Access Manager are a complex deployment on their own. This figure represents those components by a “cloud.”

Figure 28-1 Communications Services Single-tiered Deployment Example for One Host



The following table explains the protocols and port numbers used by this deployment.

Table 28-1 Protocols And Ports Used by Single-tiered Deployment Example

Protocol	Port	Used By
SMTP	25	Messaging Server MTA component to communicate with other systems, and Calendar Server (`csenpd`) component to send email notifications
HTTP	80	Clients to communicate with the Messaging Server Webmail (`httpd`) components
HTTP	81	Clients to communicate with Calendar Server (`cshttpd`)
IMAP	143	Clients to communicate with the Messaging Server `imapd` components
LDAP	389	Messaging Server and Calendar Server to communicate with LDAP directory

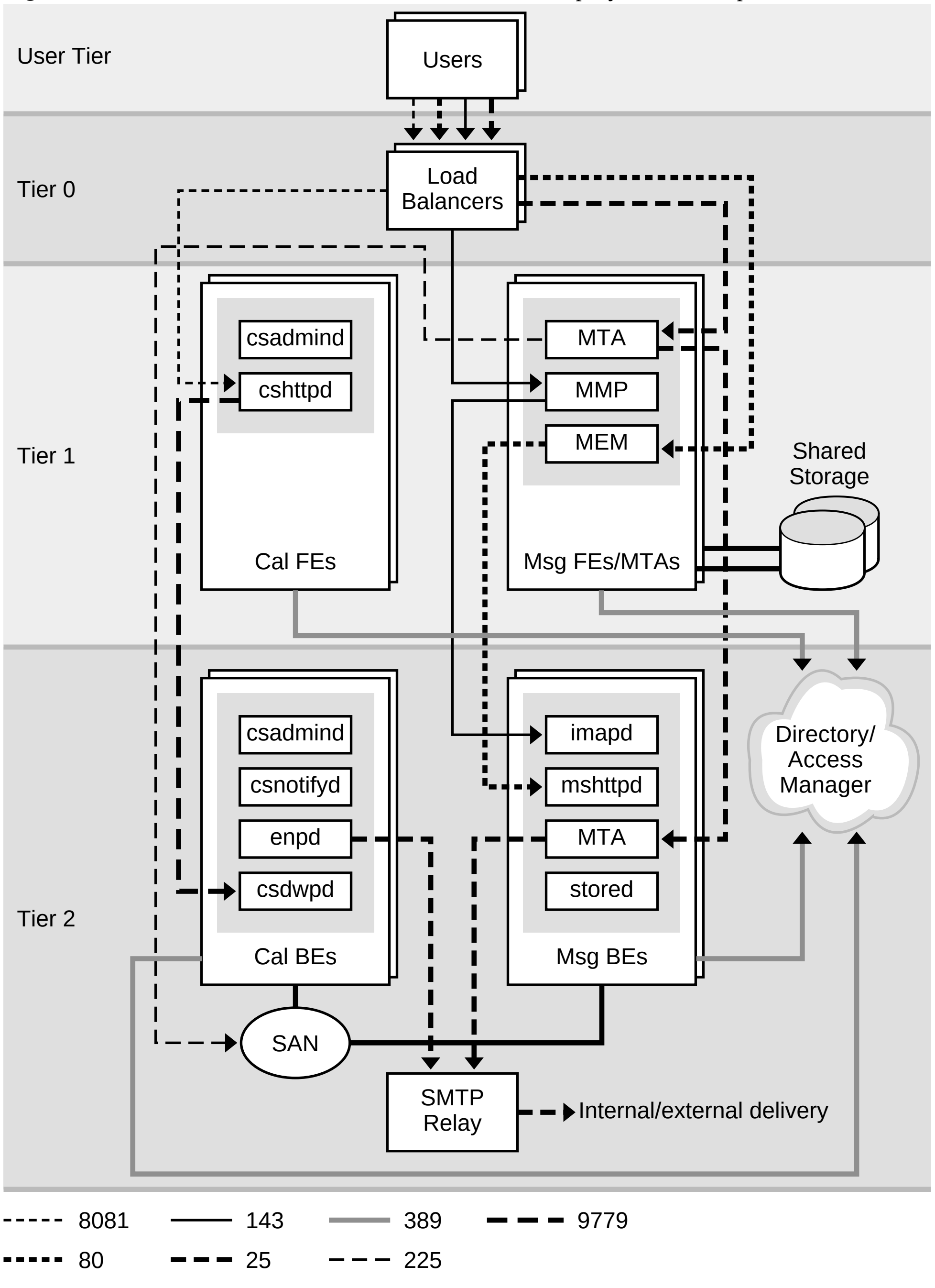
To position this deployment for future growth, you would use logical service names to install. Logical service names position deployments for easier expansion. See "Using Logical Service Names" on page 95 for more information. You would consider expanding to a two-tiered architecture when issues arise with capacity, performance, multiple geographic sites, and availability.

Communications Services Two-tiered Logical Deployment Example for Multiple Hosts

Figure 28-2 on page 359 shows a two-tiered logical deployment example for Messaging Server and Calendar Server. Tier 0 consists of load balancers. Tier 1 consists of Calendar Server and Messaging Server front ends. The Calendar Server and Messaging Server back-end stores form Tier 2.

Directory Server and Access Manager are a complex deployment on their own. This figure represents those components by a "cloud."

Figure 28-2 Communications Services Two-tiered Deployment Example



In the preceding example, load balancers form Tier 0, and direct user access to the front-end services.

The front-end services consist of four machines. Two machines are installed with Calendar Server front-end components. These Calendar Server front-end machines consist of one or two CPU servers and their own internal disk storage. Two other machines are configured as Messaging Server proxies and MTAs, and share an external disk array. These Messaging Server machines consist of four CPU servers.

The back end also consists of four machines. Two machines serve as mail stores and run the Messaging Server processes. Two other machines serve as the calendar stores and run Calendar Server process. The store machines are attached to a Storage Area Network (SAN). These back-end machines can be deployed in a variety of ways, based upon your CPU needs. Once you determine the total number of CPUs, you can opt for a vertical or horizontal configuration. For example, if your architecture called for a total of twelve CPUs, you could use three four-way servers, two six-way servers, or even one 12-way server.

Another machine serves as an SMTP relay for both Calendar Server notifications and Messaging Server emails.

The following table explains the protocols and port numbers used by this deployment.

Table 28-2 Protocols And Ports Used by Two-tiered Deployment Example

Protocol	Port	Used By
HTTP	80	Clients to communicate with the Messaging Server MEM and Webmail (`httpd`) components
SMTP	25	Clients to communicate with the Messaging Server MTA component, MTA components on the front end and back end, and Calendar Server (`csenpd`) components for email notifications
IMAP	143	Clients to communicate with the Messaging Server MMP and `imapd` components
LMTP	225	MTA routes email directly from the front end to the Message Store on the back end, bypassing the back-end MTA
LDAP	389	Front ends and back ends to communicate with LDAP directory
HTTP	8081	Clients to communicate with Calendar Front End (`cshttpd`)
DWP	9779	Calendar front end (`cshttpd`) to communicate with Calendar back end (`csdwpd`)

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 document.

Index