Sun Java Communications Suite 5 Deployment Planning Guide

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

Part No: 819–4439–11
February 2008

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more U.S. patents or pending patent applications in the U.S. and in other countries.

U.S. Government Rights – Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

This distribution may include materials developed by third parties.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, the Solaris logo, the Java Coffee Cup logo, docs.sun.com, Java, Java, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc. This product includes software developed by Computing Services at Carnegie Mellon University (`http://www.cmu.edu/computing/`).

The OPEN LOOK and Sun$^{\text{TM}}$ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

Products covered by and information contained in this publication are controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical or biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Sun Microsystems, Inc. détient les droits de propriété intellectuelle relatifs à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et ce sans limitation, ces droits de propriété intellectuelle peuvent inclure un ou plusieurs brevets américains ou des applications de brevet en attente aux Etats-Unis et dans d'autres pays.

Cette distribution peut comprendre des composants développés par des tierces personnes.

Certaines composants de ce produit peuvent être dérivées du logiciel Berkeley BSD, licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays; elle est licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, le logo Solaris, le logo Java Coffee Cup, docs.sun.com, Java, Java et Solaris sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc. Ce produit comprend du logiciel dévelopé par Computing Services à Carnegie Mellon University (`http://www.cmu.edu/computing/`).

L'interface d'utilisation graphique OPEN LOOK et Sun a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui, en outre, se conforment aux licences écrites de Sun.

Les produits qui font l'objet de cette publication et les informations qu'il contient sont régis par la legislation américaine en matière de contrôle des exportations et peuvent être soumis au droit d'autres pays dans le domaine des exportations et importations. Les utilisations finales, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes chimiques ou biologiques ou pour le nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers des pays sous embargo des Etats-Unis, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exclusive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la legislation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement designés, sont rigoureusement interdites.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFACON.

080206@19028

Contents

Preface 19

Part I **Deployment Planning Overview** 27

1 **Introduction to Deploying Communications Suite** 29

Communications Suite Overview 29

About Messaging Server 30

About Calendar Server 31

About Instant Messaging 31

About Communications Express 32

About Communications Sync 32

About Connector for Microsoft Outlook 32

Communications Suite Component Product Dependencies 33

How Communications Suite Satisfies Business Needs 33

How Messaging Server Satisfies Business Needs 33

How Calendar Server Satisfies Business Needs 34

How Instant Messaging Satisfies Business Needs 34

How Communications Express Satisfies Business Needs 34

Summary of Communications Suite Benefits 34

Making the Communications Suite Deployment Highly Available 35

Using Portal Server with Communications Suite 36

Understanding the Deployment Process 37

Analyzing Business Requirements 37

Analyzing Technical Requirements 37

Designing the Logical Architecture 38

Designing the Deployment Architecture 39

Implementing the Deployment 39

2 Analyzing Your Communications Suite Requirements 41
Identifying Deployment Goals 41
Defining Business Requirements 42
Defining Technical Requirements 42
Defining Financial Requirements 44
Defining Service Level Agreements (SLAs) 44
Determining Project Goals 45
Planning for Growth 45

3 Understanding Product Requirements and Considerations 47
Planning for Various Components 47
Understanding Service Components and Service Tiers 48
LDAP Directory Information Tree Requirements 50
Changes in the DIT Structure 50
Benefits of a One-Tree DIT Structure 51
Schema Requirements 53
Directory Server Considerations 54
Directory Server and Tiered Architecture Considerations 55
Directory Server Topology Considerations 55
Directory Server Capacity Planning 55
Directory Server and Calendar Server Interaction Considerations 56
Directory Server and Personal Address Book Considerations 56
Messaging Server Considerations 57
Calendar Server Considerations 57
Instant Messaging Considerations 59
Portal Server Considerations 59
Connector for Microsoft Outlook Considerations 60
Connector for Microsoft Outlook Component Product Dependencies 60
Migrating Sun ONE Calendar Server Data 61
Migrating Exchange Server Data 61
Communications Express Considerations 61
S/MIME Considerations 61

4 Determining Your Network Infrastructure Needs 63
Understanding Your Existing Network 63

Understanding Network Infrastructure Components 64
Routers and Switches 64
Firewall Access Control 64
Load Balancers 65
Storage Area Networks (SANs) 65
Domain Name System (DNS) 66
Planning Your Network Infrastructure Layout 66
Demilitarized Zone (DMZ) 66
Intranet 67
Internal Network 67
Proxy Hosts 68
Firewall Configuration 68
Mobile Users 68

5 Developing a Communications Suite Logical Architecture 69
Communications Suite Deployment Logical Architectures Overview 69
Single-tiered Logical Architecture for One Host 70
Single-tiered Logical Architecture for Multiple Hosts 71
Single-tiered Distributed Logical Architecture 72
Two-tiered Logical Architecture 73
Edge Logical Architecture 75
Benefits of a Single-tiered Architecture 77
Benefits of a Two-tiered Architecture 77
Horizontal Scalability Strategy 79
Scaling Front-end and Back-end Services 79
Other Deployment Issues 80
Implementing Local Message Transfer Protocol (LMTP) for Messaging Server 80
Implementing Realtime Blackhole List (RBL) 80
Using Logical Service Names 81

6 Designing for Service Availability 83
High Availability Solutions Overview 83
Automatic System Reconfiguration (ASR) 84
Directory Server and High Availability 84
Application Server and High Availability 85

Messaging Server and Calendar Server and High Availability ... 86
Instant Messaging and High Availability ... 86
Instant Messaging High Availability Overview ... 86
Using Multiple Instant Messaging Multiplexors ... 87
Using the Instant Messaging Watchdog Process ... 87
Using Enabling Techniques and Technologies ... 87
Using Load Balancers ... 88
Using Directory Proxy Server ... 88
Using Replica Role Promotion ... 88
Locating High Availability Product Reference Information ... 89
Understanding Remote Site Failover ... 89
Questions for Remote Site Failover ... 91

7 Designing for Security ... 93
Communications Suite Security Overview ... 93
Creating a Security Strategy ... 94
Physical Security ... 95
Server Security ... 95
Operating System Security ... 95
Network Security ... 96
Messaging Security ... 96
Application Security ... 97
Understanding Security Misconceptions ... 99
Other Security Resources ... 100

8 Understanding Schema and Provisioning Options ... 101
Understanding Schema Choices ... 101
Understanding Messaging Server Schema Choices ... 101
Understanding Calendar Server Schema Choices ... 103
Understanding Provisioning Tools ... 106
Understanding Messaging Server Provisioning Tools ... 106
Understanding Calendar Server Provisioning Tools ... 109

Part II **Deploying Messaging Server** 113

9 **Introduction to Messaging Server Software** 115
What Is a Messaging System? 115
Messaging Server Support for Standards and Functionality 116
Support for Standard Protocols 116
Support for Hosted Domains 116
Support for User Provisioning 117
Support for Unified Messaging 118
Support for Webmail 118
Messaging Server Security and Access Control 118
Messaging Server Administration User Interfaces 118
Messaging Server Software Architecture 119
Message Path Through the Simplified Messaging Server System 120
The Message Transfer Agent (MTA) 121
The Message Store 125
Messaging Server and Directory Services 127
Provisioning Messaging Users 128
Message Archiving Using the AXS-One System 128

10 **Planning a Messaging Server Sizing Strategy** 129
Collecting Messaging Server Sizing Data 130
Determining Messaging Peak Volume 130
Creating Your Messaging Usage Profile 130
Defining Your Messaging User Base 134
Using a Messaging Server Load Simulator 136
▼ To Use a Load Simulator 136
Assessing Your Messaging Server System Performance 136
Messaging Server Memory Utilization 136
Messaging Server Disk Throughput 137
Messaging Server Disk Capacity 137
Disk Sizing for MTA Message Queues 138
Messaging Server Network Throughput 140
Messaging Server CPU Resources 141
Developing Messaging Server Architectural Strategies 141

Two-tiered Messaging Server Architecture 141
▼ To Size the Message Store 142
▼ To Size Inbound and Outbound MTAs 143
▼ To Size Your MMP 143
Single-tiered Messaging Server Architecture 144
▼ To Size a Single-tiered Messaging Server Architecture 144

11 Developing a Messaging Server Architecture 147
Understanding the Two-tiered Messaging Architecture 147
Two-tiered Architecture—Messaging Data Flow 150
Understanding Horizontal and Vertical Scalability in Messaging Server 151
Planning for Horizontal Scalability 152
Planning for Vertical Scalability 155
Planning for a Highly Available Messaging Server Deployment 155
Performance Considerations for a Messaging Server Architecture 156
Message Store Performance Considerations 156
▼ To Adjust the Mailbox Database Cache Size 160
▼ To Monitor the Mailbox Database Cache Size 161
MTA Performance Considerations 162
MMP Performance Considerations 163
Messaging Server and Directory Server Performance Consideration 164

12 Designing a Messaging Server Topology 165
Identifying Your Geographic Needs 165
Designing a Messaging Topology 166
Central Topology 166
Distributed Topology 168
Hybrid Topology 170
Service Provider Topology 172
Understanding Messaging Topology Elements 173
Messaging Topology Components 173
Using MTAs to Protect Your Messaging System 174
Using MMPs 175
Using Gateways 177
Creating a Messaging Topology Example 177

Step 1: Identifying Messaging Goals ... 177
Step 2: Choosing a Topology Strategy ... 178
Step 3: Planning the Topology Elements ... 180

13 Planning Messaging Server Security ... 183
Protecting Messaging Components in Your Deployment ... 183
Protecting MTAs ... 183
Protecting the Message Store ... 191
Protecting MMPs ... 191
Planning Messaging User Authentication ... 192
Plain Text and Encrypted Password Login ... 192
Authentication with Simple Authentication and Security Layer (SASL) ... 193
Enabling Authenticated SMTP ... 194
Certificate-based Authentication with Secure Sockets Layer (SSL) ... 194
Planning Message Encryption Strategies ... 196
Encryption with SSL ... 196
Signed and Encrypted S/MIME ... 197

14 Planning a Messaging Server Anti-Spam and Anti-Virus Strategy ... 199
Anti-Spam and Anti-Virus Tools Overview ... 199
Rejecting Mail ... 200
Mailbox Filtering ... 201
Address Verification ... 201
Real-time Blackhole List ... 201
Relay Blocking ... 202
Authentication Services ... 202
Sidelining Messages ... 202
Comprehensive Tracing ... 202
Conversion Channel ... 203
Integration with Third-Party Products ... 203
Milter ... 203
Anti-Spam and Anti-Virus Considerations ... 204
Architecture Issues with Anti-Spam and Anti-Virus Deployments ... 204
Security Issues with Anti-Spam and Anti-Virus Deployments ... 205
Implementing an RBL ... 206

Common Anti-Spam and Anti-Virus Deployment Scenarios 206
Using Symantec Brightmail 206
Using SpamAssassin 206
Using Symantec AntiVirus Scan Engine (SAVSE) 207
Developing an Anti-Spam and Anti-Virus Site Policy 207

Part III Deploying Calendar Server 209

15 Introduction to Calendar Server Software 211
Calendar Server Overview 211
Designing Your Calendar Server Deployment 213
Objectives of Your Calendar Server Deployment 213
Calendar Server Deployment Team 214
Calendar Server End Users 214
Expected Calendar Server End User Performance 215

16 Developing a Calendar Server Architecture 217
Single-Server Calendar Server Architecture 217
Two-tiered Calendar Server Architecture 219
Two-tiered, Multiple Server Calendar Server Architecture 221

17 Planning Calendar Server Security 225
Calendar Server Security Overview 225
Monitoring Your Security Strategy 226
Planning Calendar User Authentication 226
Plain Text and Encrypted Password Login 226
Certificate-based Authentication with Secure Sockets Layer (SSL) 227

18 Planning Calendar Server Services 229
Planning for Calendar Server Front-end and Back-end Services 229
Planning for the Calendar Server LDAP Data Cache 231
Considerations for 231
Using the LDAP Data Cache in a Master/Slave LDAP Configuration 232
LDAP Attributes Affected by Delays 232

Resolving the Master-Slave Delay Problem ... 233
Configuring the LDAP Data Cache ... 234
Planning Calendar Server Domains ... 234

Part IV Deploying Instant Messaging ... 235

19 Introduction to Instant Messaging Software ... 237
What Is an Instant Messaging Service? ... 237
Instant Messaging Core Product Components ... 238
Components Related to Instant Messaging ... 238
Web Container ... 239
LDAP Server ... 239
SMTP Server ... 240
Calendar Server ... 240
Access Manager and Access Manager SDK ... 240
Portal Server ... 240
Instant Messaging Supported Standards ... 241
Instant Message Structure Format ... 241
Instant Messaging Software Architecture ... 242
Instant Messaging Server ... 244
Instant Messaging Multiplexor ... 245
Instant Messenger Client ... 245
Designing Your Instant Messaging Deployment ... 246

20 Planning an Instant Messaging Sizing Strategy ... 249
Instant Messaging Sizing Strategy Overview ... 249
Collecting Instant Messaging Sizing Data ... 250
Determining Peak Volume of Unique Instant Messaging Logins ... 250
Creating Your Instant Messaging Usage Profile ... 250
Defining Your Instant Messaging User Base or Site Profile ... 253
Using an Instant Messaging Load Simulator ... 254
Understanding Instant Messaging System Performance Guidelines ... 254
Instant Messaging Memory Utilization ... 254
Instant Messaging Disk Throughput ... 255

Instant Messaging Disk Capacity 255
Instant Messaging Network Throughput 256
Instant Messaging CPU Resources 256
Instant Messaging Multiplexor Configuration Best Practices 257
Developing Instant Messaging Architectural Strategies 257
Two-tiered Instant Messaging Architecture 257
One-tiered Instant Messaging Architecture 259
Using Load Balancers With Instant Messaging 259
Example Instant Messaging Resource Requirements 260
Small Deployment Sample Resource Requirements Numbers 260
Large Deployment Sample Resource Requirements Numbers 260

21 Developing an Instant Messaging Architecture 261
Basic Instant Messaging Architecture 261
Authentication in a Basic Architecture 263
Instant Messaging Email Notification (Calendar Alert) Architecture 265
Instant Messaging Access Manager or SSO Architecture 268
Authentication in an Access Manager Only Architecture 269
Instant Messaging Portal-based or Archiving Architecture 271
Authentication in a Portal Server Architecture 273
Instant Messaging With All Features Enabled 275
Instant Messaging Physical Deployment Examples 275
Instant Messaging Physical Deployment Example: Web Server on Separate Host 276
Instant Messaging Physical Deployment Example: Multiplexors on Separate Hosts 276
Instant Messaging Physical Deployment Example: Multiple Instant Messaging Hosts 277

22 Planning Instant Messaging Security 279
Protecting Instant Messaging Components in Your Deployment 279
Overview of Instant Messaging Security 279
Protecting Instant Messaging Server and Multiplexor 280
Providing Instant Messaging Client Access Through a Firewall 281
Protecting the Instant Messaging Archive 282
Planning Instant Messaging User Authentication 282
Instant Messaging and Passwords 282
Instant Messaging and LDAP 282

Instant Messaging and Searching the Directory Anonymously 283
Planning Instant Message Privacy, Security, and Site Policies 283
Instant Messaging Site Policies 283
Methods of Controlling Instant Messaging End User and Administrator Privileges 284

Part V Deploying Communications Express 285

23 Introduction to Communications Express Software 287
Communications Express Overview 287
Communications Express Features 288
Communications Express High-Level Architecture 288

24 Developing a Communications Express Architecture 291
Basic Communications Express Architecture 291
Communications Express on Remote Host Architecture 293

Part VI Deploying Connector for Microsoft Outlook 297

25 Introduction to Connector for Microsoft Outlook Software 299
Connector for Microsoft Outlook Overview 300
Connector for Microsoft Outlook Features 300
Connector for Microsoft Outlook High-Level Architecture 302

26 Developing a Connector for Microsoft Outlook Architecture 303
Desktop Deployment Process Tasks 303
Why Use the Deployment Toolkit? 304

27 Understanding Connector for Microsoft Outlook Pre-Installation Considerations 305
Connector for Microsoft Outlook Installation Considerations 305
Interactive User Installation (Self-Service) 305
Individual Desktop Installation 306
Automated Installation Using Configuration Management Tools 306
Command-Line Switches for Desktop Installation 307

Password-Protected Personal Stores in Outlook 307
Exchange to Communications Suite Migration Considerations 308
Migration of Important Server Data: More Than Just Messages 308
Mail Routing During Transition Phase of Server Migration 308
Synchronization of Global Address Books During Migration 309
New User IDs and Passwords for Sun Java System Servers 309
Where to Go for More Connector for Microsoft Outlook Information 310

Part VII Deployment Examples 311

28 Communications Suite Deployment Examples 313
Communications Suite Single-tiered Logical Deployment Example for One Host 313
Communications Suite Two-tiered Logical Deployment Example for Multiple Hosts 316

Glossary 319

Index 321

Tables

TABLE 1–1 How Communications Suite Benefits Your Organization 35

TABLE 2–1 Considerations for Total Cost of Ownership 46

TABLE 5–1 User Facing Logical Names 81

TABLE 5–2 Maintenance Level Logical Names 81

TABLE 5–3 Mapping of User Level to Maintenance Level Logical Names 82

TABLE 6–1 Designing Directory Server for High Availability 85

TABLE 8–1 Messaging Server Provisioning Mechanisms 108

TABLE 8–2 Calendar Server Provisioning Mechanisms 111

TABLE 11–1 High Access Messaging Server Directories 157

TABLE 13–1 Common MTA Security Threats 184

TABLE 13–2 Access Control Mapping Tables 185

TABLE 13–3 SASL Authentication User Access Protocols Support Matrix 193

TABLE 13–4 SSL Authentication Support Matrix 195

TABLE 18–1 Calendar Server LDAP Attributes Affected by Delays 232

TABLE 20–1 Instant Messaging Server and Multiplexor Memory Disk Space Sizing for Concurrent Users 256

TABLE 20–2 Instant Messaging CPU Utilization Numbers 257

TABLE 24–1 Protocols And Ports Used by Basic Communications Express Deployment Architecture 292

TABLE 24–2 Protocols And Ports Used by Communications Express Remote Host Deployment Example 295

TABLE 28–1 Protocols And Ports Used by Single-tiered Deployment Example 316

TABLE 28–2 Protocols And Ports Used by Two-tiered Deployment Example 318

Figures

FIGURE 3–1 Communications Suite Components 49

FIGURE 3–2 Two-Tree LDAP Structure Compared With One-Tree Structure 51

FIGURE 3–3 Two-Tree Aliasing With `aliasedDomainName` and `inetDomainBaseDN` 52

FIGURE 3–4 Two-Tree Aliasing With `inetCanonicalDomainName` 52

FIGURE 3–5 One-Tree Aliasing With `associatedDomain` 53

FIGURE 5–1 Single-tiered Architecture for One Host 70

FIGURE 5–2 Single-tiered Architecture for Multiple Hosts 71

FIGURE 5–3 Single-tiered Distributed Architecture 72

FIGURE 5–4 Two-tiered Architecture 74

FIGURE 5–5 Edge Architecture 76

FIGURE 9–1 Standalone Messaging Server, Simplified Components View 119

FIGURE 9–2 Channel Architecture 122

FIGURE 10–1 Simplified Messaging Server Two-tiered Architecture 142

FIGURE 10–2 Simplified Messaging Server Single-tiered Architecture 144

FIGURE 11–1 Two-Tiered Messaging Server Architecture 148

FIGURE 11–2 Spreading Your User Base Across Multiple Servers 153

FIGURE 12–1 Central Topology 167

FIGURE 12–2 Distributed Topology 169

FIGURE 12–3 Hybrid Topology 171

FIGURE 12–4 Service Provider Topology 172

FIGURE 12–5 MTAs in Messaging Topology 175

FIGURE 12–6 MMP Overview 176

FIGURE 12–7 Hybrid Topology for the Siroe Corporation 179

FIGURE 12–8 Topological Elements in the Siroe Messaging Deployment for Chicago and Minneapolis 180

FIGURE 13–1 Mapping Tables and the Mail Acceptance Process 186

FIGURE 14–1 Incorrect Deployment of Anti-Spam/Virus Solution 205

FIGURE 14–2 Correct Deployment of Anti-Spam/Virus Solution 205

FIGURE 16–1 Single-Server Calendar Server Architecture 218

FIGURE 16–2 Two-tiered Calendar Server Architecture 220
FIGURE 16–3 Two-tiered, Multiple Server Calendar Server Architecture 222
FIGURE 19–1 Instant Messaging Software Architecture 243
FIGURE 20–1 Simplified Two-tiered Instant Messaging Architecture 258
FIGURE 20–2 Simplified One-tiered Instant Messaging Architecture 259
FIGURE 21–1 Basic Instant Messaging Architecture 262
FIGURE 21–2 Flow of Authentication Requests in a Basic Instant Messaging Architecture .. 264
FIGURE 21–3 Instant Messaging Architecture with Email Notification 266
FIGURE 21–4 Instant Messaging Architecture with Calendar Alerts 267
FIGURE 21–5 Instant Messaging Architecture With Access Manager-based Server Policy Management or Single Sign On 268
FIGURE 21–6 Flow of Authentication Requests in an Access Manager Configuration 270
FIGURE 21–7 Instant Messaging Architecture With Portal-based Secure Mode or Archiving 272
FIGURE 21–8 Flow of Authentication Requests in a Portal Server and Access Manager Configuration 274
FIGURE 21–9 Separate Web Server and Instant Messaging Hosts 276
FIGURE 21–10 Instant Messaging Multiplexors Installed on Separate Hosts 277
FIGURE 21–11 Multiple Instant Messaging Server Hosts 278
FIGURE 23–1 High Level Communications Express Software Architecture 289
FIGURE 24–1 Basic Communications Express Architecture 292
FIGURE 24–2 Communications Express on Remote Host Architecture 294
FIGURE 28–1 Communications Suite Single-tiered Deployment Example for One Host 315
FIGURE 28–2 Communications Suite Two-tiered Deployment Example 317

Preface

The *Sun Java Communications Suite 5 Deployment Planning Guide* contains the information you need to deploy Sun Java™ Communications Suite 5. This guide helps you through the process of understanding Sun Java Communications Suite, evaluating and analyzing your site, and designing the kind of deployment architecture that meets your organization's needs.

Who Should Use This Book

This guide is for individuals who are responsible for assessing and deploying Communications Suite 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, LDAP, and XMPP 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 Suite 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 detailed information on deploying the Connector for Microsoft Outlook. Part VII provides deployment examples. The following table summarizes the content of this book.

TABLE P–1 How This Book Is Organized

Chapter	Description
Chapter 1, "Introduction to Deploying Communications Suite,"	Provides an overview of Communications Suite.
Chapter 2, "Analyzing Your Communications Suite Requirements,"	Explains how to analyze your organization's business and technical requirements.
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 Suite Logical Architecture,"	Describes how to develop your Communications Suite 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 Suite.
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, "Introduction to Calendar Server Software,"	Provides an overview of the Calendar Server software.
Chapter 16, "Developing a Calendar Server Architecture,"	Describes basic architectures of your Calendar Server deployment.

TABLE P–1 How This Book Is Organized *(Continued)*

Chapter	Description
Chapter 17, "Planning Calendar Server Security,"	Describes how to plan for and protect the various components of your Calendar Server deployment.
Chapter 18, "Planning Calendar Server Services,"	Describes the additional considerations of a Calendar Server deployment, with respect to Calendar Server services.
Chapter 19, "Introduction to Instant Messaging Software,"	Provides an overview of the Instant Messaging software.
Chapter 20, "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 21, "Developing an Instant Messaging Architecture,"	Describes a variety of Instant Messaging architectures.
Chapter 23, "Introduction to Communications Express Software,"	Provides an overview of the Communications Express software.
Chapter 24, "Developing a Communications Express Architecture,"	Describes basic Communications Express architectures.
Chapter 25, "Introduction to Connector for Microsoft Outlook Software,"	Provides an overview of the Connector for Microsoft Outlook software.
Chapter 26, "Developing a Connector for Microsoft Outlook Architecture,"	Describes basic architectures of your Connector for Microsoft Outlook deployment.
Chapter 27, "Understanding Connector for Microsoft Outlook Pre-Installation Considerations,"	Describes considerations you need to think about before installing Connector for Microsoft Outlook.
Chapter 28, "Communications Suite Deployment Examples,"	Provides Communications Suite deployment examples.
Glossary	Provides a link to the Java Enterprise System glossary.

Communications Suite Documentation Set

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

TABLE P–2 Communications Suite Documentation

Document Title	Contents
Sun Java Communications Suite 5 Schema Reference	Serves as a reference for schema information for Sun Java Communications Suite products using LDAP, specifically Messaging Server and Calendar Server.
Sun Java System Communications Services 6 2005Q4 Schema Migration Guide	Describes how to migrate Sun Java System LDAP Directory data from LDAP Schema 1 to LDAP Schema 2 for Sun Java Communications Suite, specifically Messaging Server and Calendar Server.
Sun Java System Delegated Administrator 6.4 Administration Guide	Describes how to configure and administer Sun Java System Delegated Administrator.

Related Books

For other server documentation related to deploying Communications Suite, go to the following:

- Access Manager documentation: `http://docs.sun.com/app/docs/coll/1292.2`
- Calendar Server documentation: `http://docs.sun.com/app/docs/coll/1313.2`
- Communications Express documentation: `http://docs.sun.com/app/docs/coll/1312.2`
- Directory Server documentation: `http://docs.sun.com/app/docs/coll/1316.2`
- Instant Messaging documentation: `http://docs.sun.com/app/docs/coll/1309.3`
- Messaging Server documentation: `http://docs.sun.com/app/docs/coll/1312.2`

Default Paths and File Names

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

TABLE P–3 Default Path and File Name

Placeholder	Description	Default Value
product-base	Represents the base installation directory for Messaging Server. The Messaging Server 6.3 default base installation and product directory depends on your specific platform.	Solaris systems: `/opt/SUNWmgsr` Linux systems: `/opt/sun/messaging`

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–4 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	A placeholder to be replaced with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized (note that some emphasized items appear bold online)	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file.

Shell Prompts in Command Examples

The following table shows default system prompts and superuser prompts.

TABLE P–5 Shell Prompts

Shell	Prompt
C shell on UNIX and Linux systems	`machine_name%`
C shell superuser on UNIX and Linux systems	`machine_name#`
Bourne shell and Korn shell on UNIX and Linux systems	`$`
Bourne shell and Korn shell superuser on UNIX and Linux systems	`#`
Microsoft Windows command line	`C:\`

Symbol Conventions

The following table explains symbols that might be used in this book.

TABLE P–6 Symbol Conventions

Symbol	Description	Example	Meaning
`[ ]`	Contains optional arguments and 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.
`${ }`	Indicates a variable reference.	`${com.sun.javaRoot}`	References the value of the `com.sun.javaRoot` variable.
-	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.
→	Indicates menu item selection in a graphical user interface.	File → New → Templates	From the File menu, choose New. From the New submenu, choose Templates.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused 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 full document title and part number. The part number is a 7-digit or 9-digit number that can be found on the book's title page or in the document's URL. For example, the part number of this book is 819-4439.

Revision History

TABLE P–7 Revision History

Version	Date	Description of Changes
10	March 2007	Initial release of this document.
11	February 2008	Fixes for CRs 6646117 and 6581001. Updated "Communications Express on Remote Host Architecture" on page 293.

PART I

Deployment Planning Overview

This part contains the following chapters:

- Chapter 1, “Introduction to Deploying Communications Suite,”
- Chapter 2, “Analyzing Your Communications Suite Requirements,”
- Chapter 3, “Understanding Product Requirements and Considerations,”
- Chapter 4, “Determining Your Network Infrastructure Needs,”
- Chapter 5, “Developing a Communications Suite Logical Architecture,”
- Chapter 6, “Designing for Service Availability,”
- Chapter 7, “Designing for Security,”
- Chapter 8, “Understanding Schema and Provisioning Options,”

◆ ◆ ◆ CHAPTER 1

Introduction to Deploying Communications Suite

This chapter provides an overview of Sun Java Communications Suite 5, the business reasoning behind deploying Communications Suite, and the deployment process itself.

This chapter contains the following sections:

- "Communications Suite Overview" on page 29
- "How Communications Suite Satisfies Business Needs" on page 33
- "Understanding the Deployment Process" on page 37

Communications Suite Overview

Sun Java Communications Suite 5 is a secure, cost-effective communications and collaborations offering. Communications Suite addresses 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 Suite provides 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 Suite 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 Suite enables organizations to deploy solutions at a total cost of ownership they can afford.

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

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

- Sun Java System Messaging Server 6.3 (formerly Sun™ ONE Messaging Server)
- Sun Java System Calendar Server 6.3 (formerly Sun ONE Calendar Server)
- Sun Java System Instant Messaging 7.2 (formerly Sun ONE Instant Messaging)

Additional features that enhance the Communications Suite solution include:

- Sun Java System Communications Express 6.3
- Sun Java System Communications Sync 3.0
- Sun Java System Connector for Microsoft Outlook 7.2

Taken as a whole, Communications Suite provides 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 Suite delivers a robust and flexible platform meeting the diverse communications needs of all types of organizations. Communications Suite is an optimal solution to connect remote offices, distributed workgroups, and global corporate locations.

About Messaging Server

Sun Java System Messaging Server 6.3 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 the total cost of ownership of the 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. Messaging Server also can provide an archive for instant messaging communications. Furthermore, Messaging Server supports archiving through the AXS-One message archiving system to provide an industrial strength message archiving solution for both compliance and operational archiving.

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

Messaging Server currently ships with two native 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 for more information on Messaging Server concepts and other deployment aspects.

About Calendar Server

Sun Java System Calendar Server 6.3 facilitates team collaboration by enabling users to manage and coordinate appointments, events, tasks, and resources. Through an integrated Web interface or connectors to other calendar clients (including Microsoft Outlook), Calendar Server enables end users to access their personal, public, or group calendars anytime from 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 ships with two native 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 time line for Calendar Express at a future date.

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

About Instant Messaging

Sun Java System Instant Messaging 7.2 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 and Sun Java System Access Manager 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, gateways, and components. Some of these clients 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 for more information on Instant Messaging concepts and deployment aspects.

About Communications Express

Sun Java System Communications Express 6.3 provides an integrated web-based communications and collaboration client. Communications Express is a common component between 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 for more information on Communications Express concepts and deployment aspects.

About Communications Sync

Sun Java System Communications Sync 3.0 (formerly Sun ONE Synchronization) is a software product that runs on a Windows personal computer and enables users to synchronize Communications Suite events, tasks, and contacts with mobile devices and personal information managers (PIMs) such as Microsoft Outlook.

See the Communications Sync documentation at the following location for more information:

`http://docs.sun.com/app/docs/coll/1313.2`

About Connector for Microsoft Outlook

Sun Java System Connector for Microsoft Outlook 7.2 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. Finally, 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/coll/1312.2`

Communications Suite Component Product Dependencies

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

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

How Communications Suite Satisfies 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 Suite meets these needs through providing scalable messaging, calendaring, and instant messaging at a total cost of ownership businesses can afford.

Communications Suite enables you to develop an architecture that incorporates ease of deployment and maintenance with a complete set of features and functionality. Most important, a Communications Suite architecture builds security into each service element. These elements include the network infrastructure, operating environment, and the Communications Suite 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 automatically established since both applications are deployed within a single web container. Also, you can optionally use Access Manager to provide the 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 Suite Benefits

The Communications Suite 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.

The following table summarizes the benefits provided by Communications Suite.

TABLE 1–1 How Communications Suite Benefits 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	Enables access from a wide variety of clients and devices such as web-based clients, multiple rich clients including Evolution and Microsoft Outlook, and web-enabled devices such as phones and PDAs.
Open, modular, and standards-based architecture	Enables customers to deploy customized and personalized solutions.

Making the Communications Suite Deployment Highly Available

You can configure Messaging Server, Calendar Server, and Instant Messaging to be highly available by using clustering software. Messaging Server supports both Sun Cluster and Veritas Cluster Server software. Calendar Server and Instant Messaging support Sun Cluster software. When using clustering software, a secondary Messaging Server, Calendar Server, or Instant Messaging 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 processes and services automatically, if necessary. Messaging Server logs failures and recovery operations, which you can use for reporting and analysis.

Instant Messaging may make use of server pools to provide redundancy. In a server pool, two or more Instant Messaging servers provide the instant messaging services for a single domain. If a server fails, another server in the pool handles the additional load.

Additionally, you can deploy the Communications Suite 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 Suite. See the Sun Cluster, Messaging Server, Calendar Server, and Instant Messaging documentation for more information on this topic.

Using Portal Server with Communications Suite

You can install Communications Suite products with Portal Server to provide access to messaging and calendar services (by creating portlets in a portal page). For example, you can create these portlets to 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 Suite deployment:

- **Portal Server Desktop.** Enables users to access and launch Communications Suite applications from portlets that you create.

- **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 deploying Communications Suite in a portal environment. See the Portal Server documentation for more information.

Understanding the Deployment Process

The Communications Suite 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 Suite, and Java Enterprise System components, see the *Sun Java Enterprise System Deployment Planning Guide*.

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 6.3 WCAP Developer's Guide*
- *Sun Java Communications Suite 5 Event Notification Service Guide*
- *Sun Java System Messenger Express 6 2005Q4 Customization Guide*

- *Sun Java System Messaging Server 6 2005Q4 MTA Developer's Reference*

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 Suite Requirements

Planning your Communications Suite 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 Suite design.

This chapter contains the following sections:

- "Identifying Deployment Goals" on page 41
- "Determining Project Goals" on page 45

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

Identifying Deployment Goals

Before you purchase or deploy Communications Suite 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 Requirements

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 Suite services themselves. See "Using Logical Service Names" on page 81 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 Suite. 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 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 Suite architecture.

This chapter contains the following sections:

- "Planning for Various Components" on page 47
- "LDAP Directory Information Tree Requirements" on page 50
- "Schema Requirements" on page 53
- "Directory Server Considerations" on page 54
- "Messaging Server Considerations" on page 57
- "Calendar Server Considerations" on page 57
- "Instant Messaging Considerations" on page 59
- "Portal Server Considerations" on page 59
- "Connector for Microsoft Outlook Considerations" on page 60
- "Communications Express Considerations" on page 61

Planning for Various Components

When designing your Communications Suite 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 Suite with other Java System products, you need to choose your schema accordingly. Inter-product dependencies, for example, how Communications Suite accesses and places 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
  - Messaging Multiplexor (MMP)
- 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 Suite deployment for multiple component products or services, you need to understand the composition of each component product (or service) itself.

Figure 3–1 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 77 and "Benefits of a Two-tiered Architecture" on page 77 for more information.

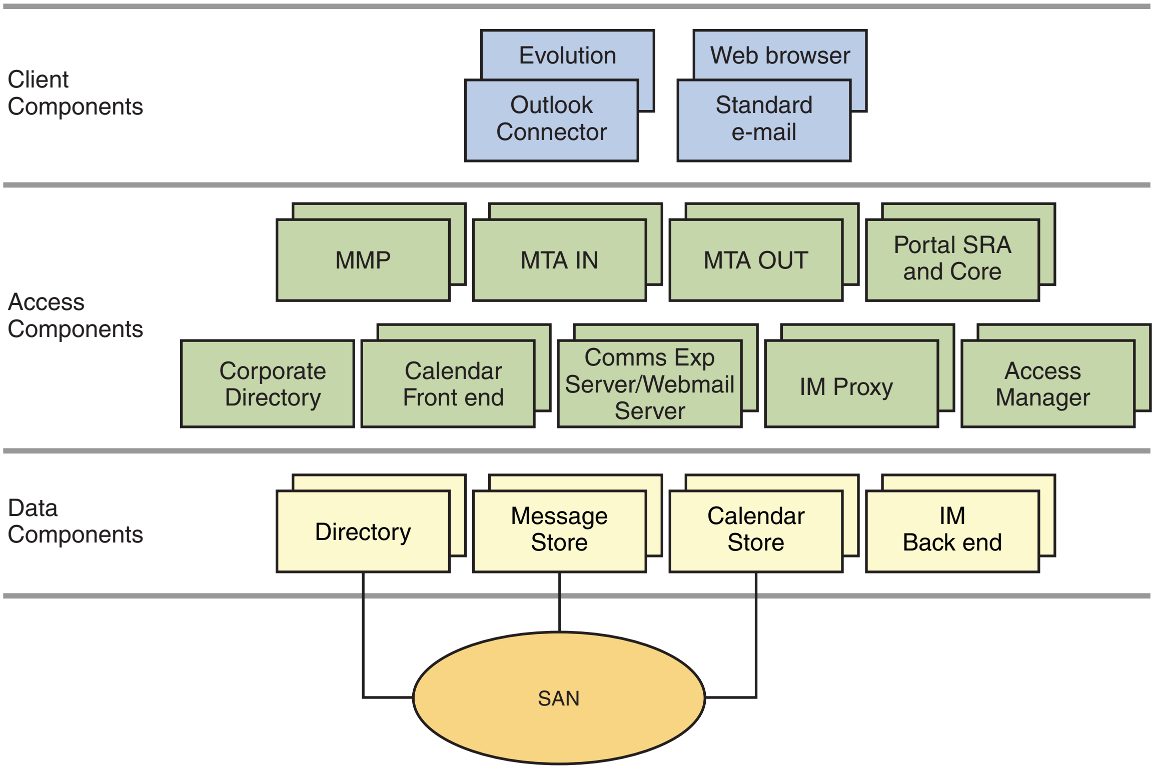

FIGURE 3–1 Communications Suite Components

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 and MTA); Calendar Server; Communications Express and Webmail Server; 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.

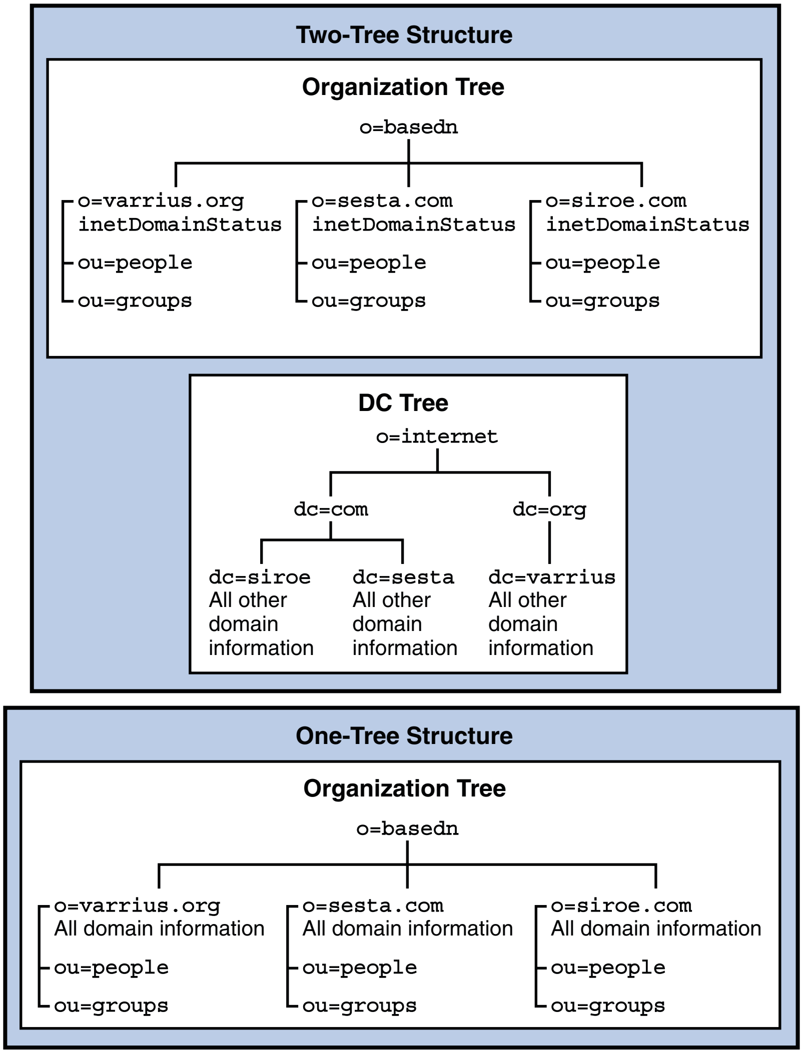

FIGURE 3–2 Two-Tree LDAP Structure Compared With One-Tree Structure

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.

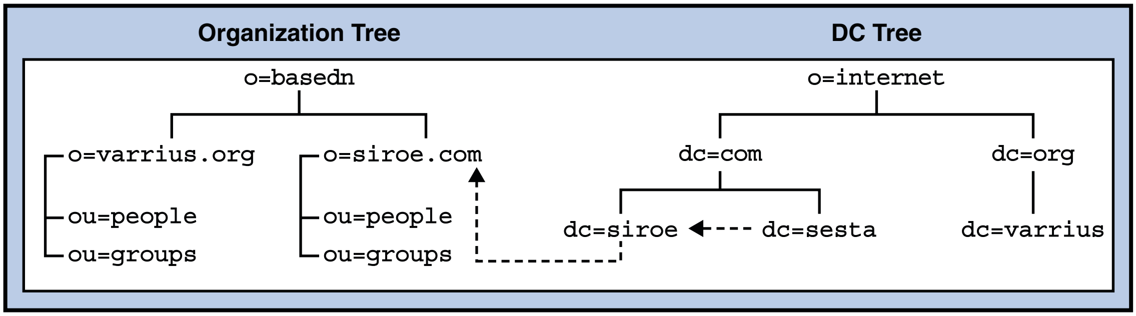

FIGURE 3–3 Two-Tree Aliasing With aliasedDomainName and inetDomainBaseDN

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, 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).

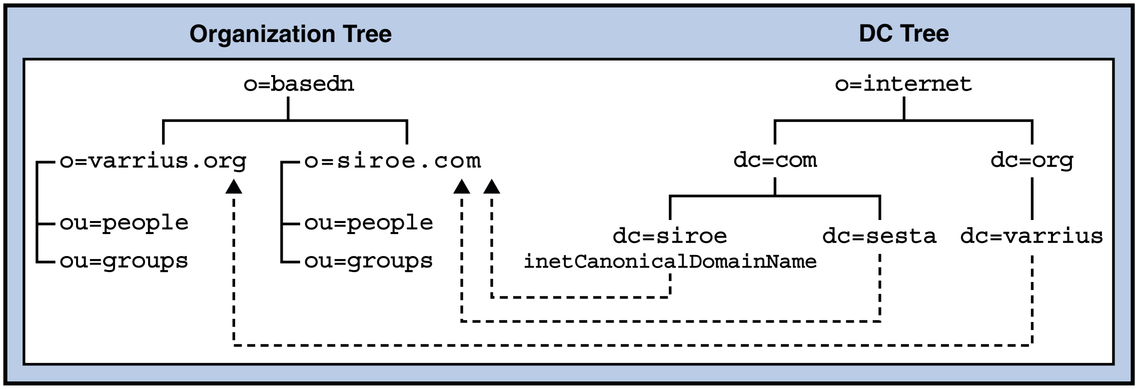

FIGURE 3–4 Two-Tree Aliasing With inetCanonicalDomainName

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

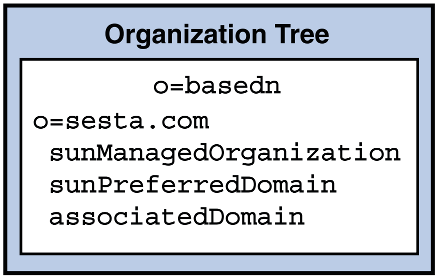

FIGURE 3–5 One-Tree Aliasing With `associatedDomain`

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 Suite 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 Suite. Your choice of schema depends on the following criteria:

Use Schema 2 if you:

- Want to use Access Manager 6 (formerly Identity Server), either for user provisioning or single sign-on (SSO)
- Are installing Communications Suite components for the first time
- Are integrating Communications Suite 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 Suite 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, 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 Directory Server documentation for a complete description of these factors and suggestions about how to architect your data environment:

`http://docs.sun.com/coll/1316.1`

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 Suite 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. 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 Suite 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 84 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 231 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 Suite architecture, you need to evaluate the performance aspects of the following Messaging Server components:

- Message Store
- Message Transfer Agent (MTA)
- Mail Message Proxy (MMP)

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 156.

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.

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 Calendar 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 Suite 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 257 for more information.

Portal Server Considerations

You can install Communications Suite 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 7.1 Deployment Planning Guide* and the *Sun Java System Access Manager 7.1 Deployment Planning Guide* for more information.

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/app/docs/coll/1312.2`

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 Chapter 7, “Sun Java System Connector for Microsoft Outlook 7.2 Release Notes,” in *Sun Java Communications Suite 5 Release Notes* for a complete list of product dependencies.

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” in *Sun Java Communications Suite 5 Installation Guide* 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 Suite 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" on page 63
- "Understanding Network Infrastructure Components" on page 64
- "Planning Your Network Infrastructure Layout" on page 66

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.

Firewall Access Control

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

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

In addition, when installing a Sun server in an environment that is exposed to the Internet, or any untrusted network, reduce the 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.

Proxy Hosts

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 Suite Logical Architecture

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

This chapter contains the following sections:

- "Communications Suite Deployment Logical Architectures Overview" on page 69
- "Horizontal Scalability Strategy" on page 79
- "Other Deployment Issues" on page 80

Communications Suite Deployment Logical Architectures Overview

You can deploy Communications Suite 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 Suite 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 Suite 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.

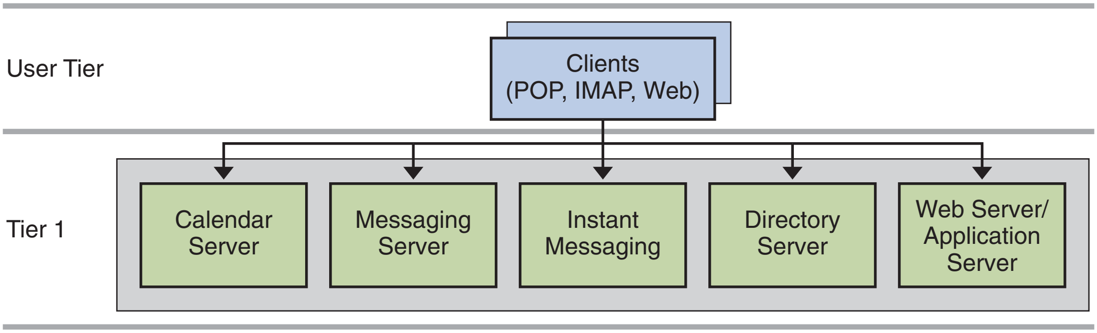

FIGURE 5–1 Single-tiered Architecture for One Host

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 81 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 represents the single-tiered logical architecture for multiple hosts.

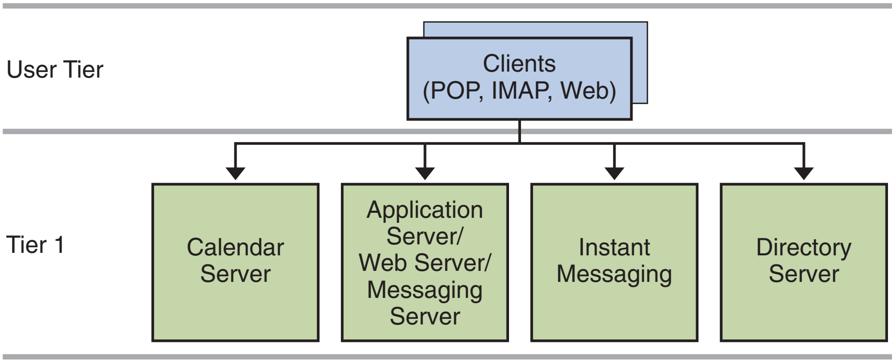

FIGURE 5–2 Single-tiered Architecture for Multiple Hosts

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 represents the single-tiered distributed logical architecture.

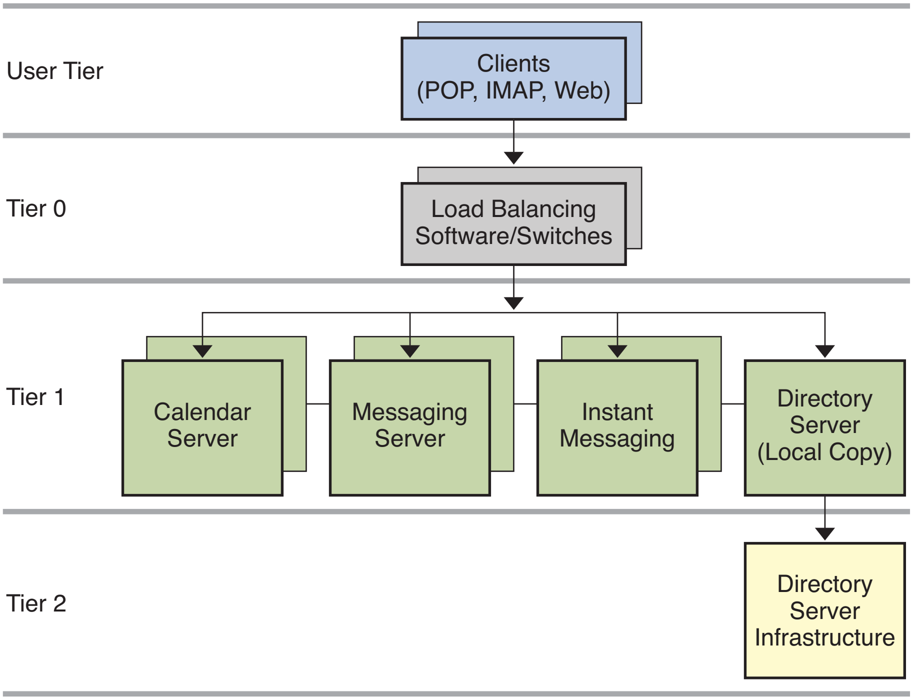

FIGURE 5–3 Single-tiered Distributed Architecture

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 Suite 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 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 represents the two-tiered logical architecture.

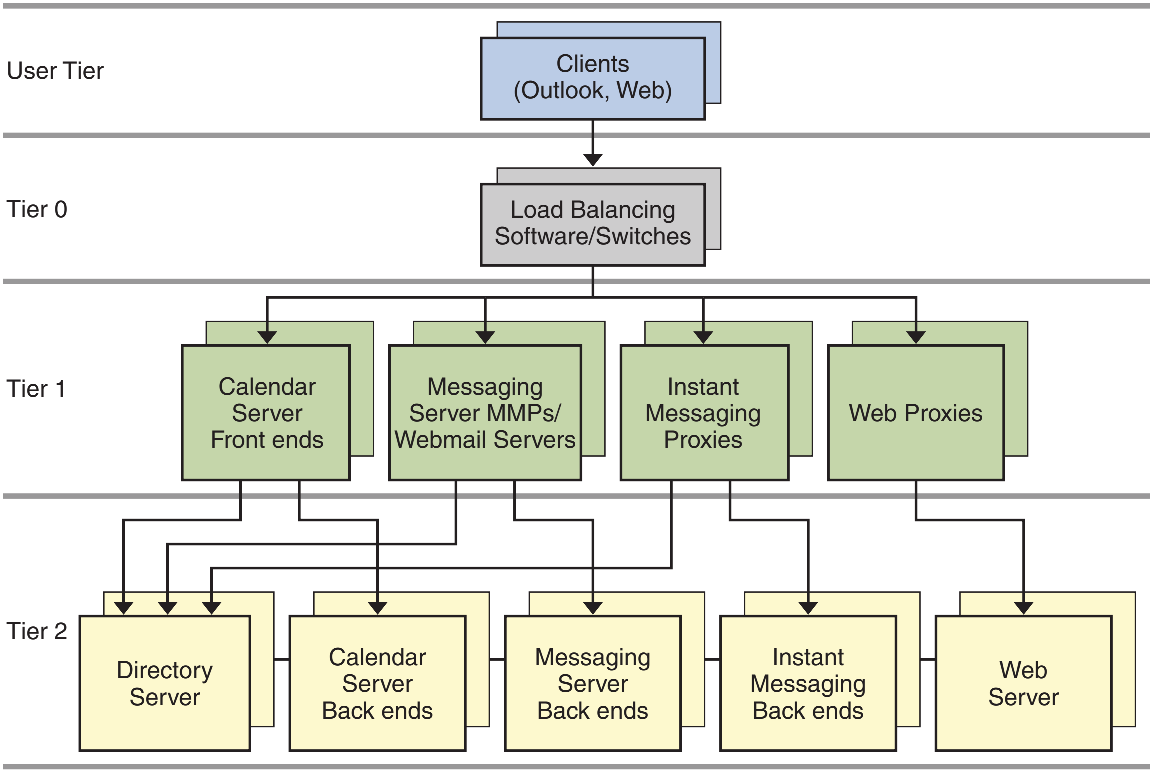
User Tier
Clients (Outlook, Web)
Tier 0
Load Balancing Software/Switches
Tier 1
Calendar Server Front ends
Messaging Server MMPs/ Webmail Servers
Instant Messaging Proxies
Web Proxies
Tier 2
Directory Server
Calendar Server Back ends
Messaging Server Back ends
Instant Messaging Back ends
Web Server

FIGURE 5–4 Two-tiered Architecture

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 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 Suite servers at every remote site.

Figure 5–5 represents the edge logical architecture.

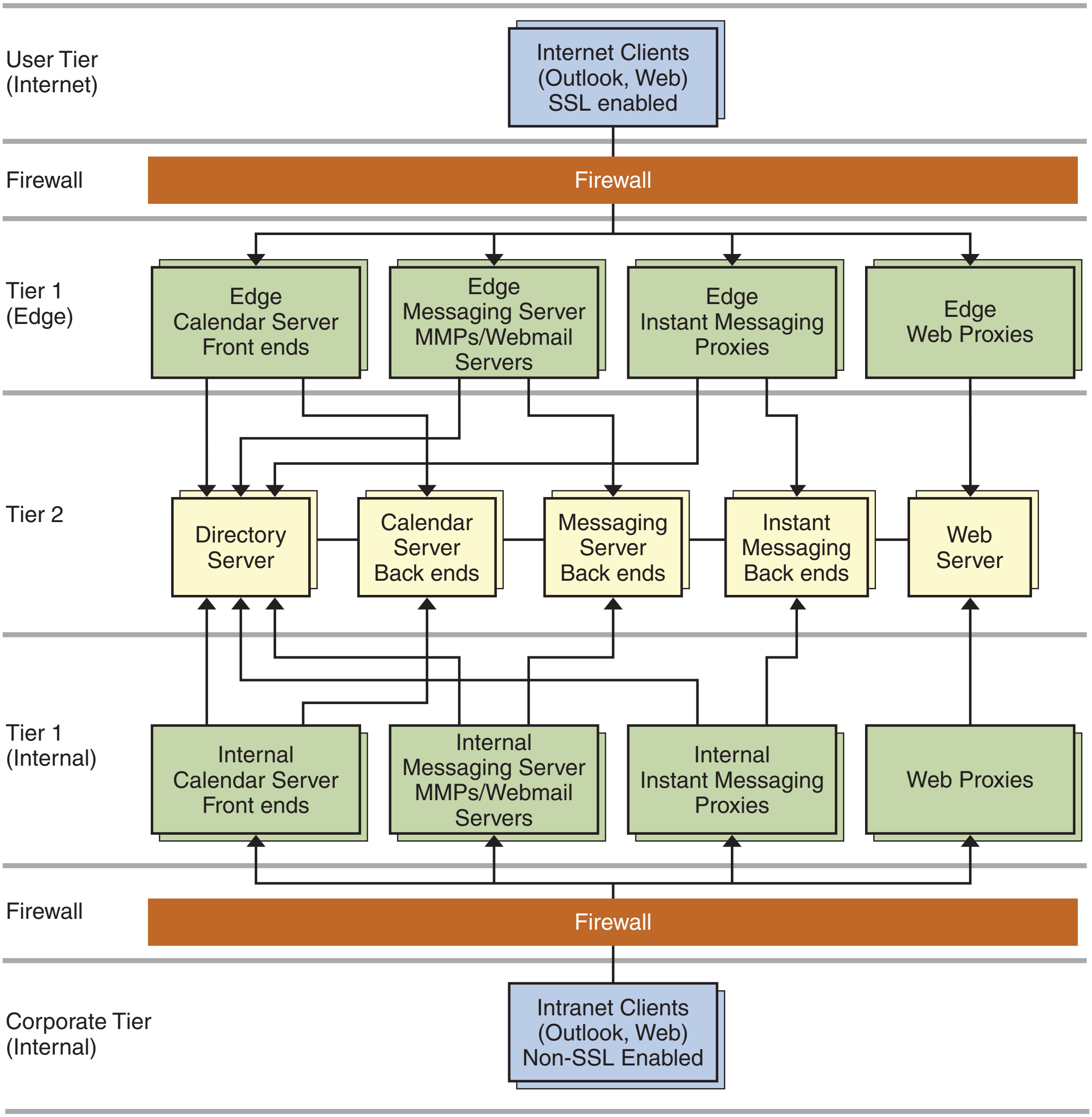

FIGURE 5–5 Edge Architecture

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 Suite 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 162), 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.** The Calendar Server front end greatly reduces 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 Suite 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 Suite 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 “To Configure LMTP Delivery” in *Sun Java System Messaging Server 6.3 Administration Guide* for instructions on configuring LMTP.

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.

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 *Sun Java System Messaging Server 6.3 Administration Reference* for information on configuring the `ENABLE_RBL` option of the MTA Dispatcher.

Using Logical Service Names

Design your deployment around the use of logical names for Communications Suite 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.

TABLE 5–2 Maintenance Level Logical Names *(Continued)*

Example	Description
`relay-out.siroe.com`	Corresponds to a bank of outbound SMTP servers.
`mmp.siroe.com`	Corresponds to a bank of MMP 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`
`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" on page 83
- "Directory Server and High Availability" on page 84
- "Application Server and High Availability" on page 85
- "Messaging Server and Calendar Server and High Availability" on page 86
- "Instant Messaging and High Availability" on page 86
- "Using Enabling Techniques and Technologies" on page 87
- "Locating High Availability Product Reference Information" on page 89
- "Understanding Remote Site Failover" on page 89

High Availability Solutions Overview

High availability solutions for Communications Suite 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 also provides a Sun Cluster agent, but it does not support VCS. You can also 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 Suite 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 Suite 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 multi-master environment between two suppliers that share responsibility for the same data, you create two replication agreements. Supplier A and Supplier B each hold a master replica of the same data and there are two replication agreements governing the replication flow of this multi-master configuration.
Four-way multi-master	Provides a pair of Directory Server masters, usually in two separate data centers. This configuration uses four-way MultiMaster Replication (MMR) for replication. Thanks to its four-way master failover configuration, this fully-connected topology provides a highly-available solution that guarantees data integrity. When used with hubs in the replication topology, load distribution is facilitated, and the four consumers in each data center allow this topology to scale for read (lookup) operations.
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 Directory Server documentation for more information:

http://docs.sun.com/coll/1316.2

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 multi-machine 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 8.2 documentation for more information:

http://docs.sun.com/app/docs/coll/1310.3

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 Suite 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 and MMP, Webmail Server, 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 Chapter 2, “Key Concepts for Hardware Service Providers,” in *Sun Cluster Concepts Guide for Solaris OS* for more information on Sun Cluster topologies.

See Chapter 3, “Configuring High Availability,” in *Sun Java System Messaging Server 6.3 Administration Guide* for more information on configuring Messaging Server for high availability.

See Chapter 6, “Configuring Calendar Server 6.3 Software for High Availability (Failover Service),” in *Sun Java System Calendar Server 6.3 Administration Guide* for more information on configuring Calendar Server for high availability.

Instant Messaging and High Availability

Instant Messaging provides a Sun Cluster agent, but it does not support Veritas Cluster Service. You can also create a “more available” environment by deploying redundant Instant Messaging multiplexors, and by taking advantage of the Instant Messaging watchdog process.

Instant Messaging High Availability Overview

Configuring Instant Messaging for high availability (HA) by using the Sun Cluster agent provides for monitoring of and recovery from software and hardware failures. The high availability feature is implemented as a failover data service, not a scalable service, and is currently supported only on Solaris.

Note – You can have multiple Instant Messaging nodes in an HA environment using the same SMTP server.

Before implementing an Instant Messaging HA environment using the Sun Cluster agent, decide which of the following HA deployments best meets your needs.

- **Mixed HA Environment.** This deployment consists of a local configuration and binaries, and global runtime files. The advantage to this setup is that upgrading Instant Messaging requires minimal downtime because you can upgrade on nodes where Instant Messaging is offline. The disadvantage is that you need to ensure that the same configuration and version of Instant Messaging exists on all nodes in the cluster. In addition, if you choose this option, you need to determine whether you will be using HAStoragePlus or the cluster file system for global runtime files.
- **Global HA Environment.** This deployment consists of a global configuration, binaries, and runtime files. This setup is easier to administer, but you need to bring Instant Messaging down on all nodes in the cluster before upgrading.

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 the Sun Cluster agent, 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/1317.3

Using Replica Role Promotion

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

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

See the Directory Server documentation for more information:

http://docs.sun.com/app/docs/coll/1310.3

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

Veritas Cluster Server: *Veritas Cluster Server User's Guide*:
`http://seer.support.veritas.com/docs/275725.htm`

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:

- "Communications Suite Security Overview" on page 93
- "Creating a Security Strategy" on page 94
- "Understanding Security Misconceptions" on page 99
- "Other Security Resources" on page 100

For detailed product security information, see Chapter 13, "Planning Messaging Server Security," Chapter 17, "Planning Calendar Server Security," and Chapter 22, "Planning Instant Messaging Security."

Communications Suite Security Overview

You manage security for a Communications Suite 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 6.3 Administration Guide*.

Your security strategy should also plan for:

- "Physical Security" on page 95
- "Server Security" on page 95
- "Operating System Security" on page 95
- "Network Security" on page 96
- "Messaging Security" on page 96
- "Application Security" on page 97

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 Suite product portfolio provides features that ensure the security and integrity of business communications. Communications Suite offers 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 Suite supports 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 Suite 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.

- **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**

  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 Suite 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 Suite. Because of the complexity in provisioning Communications Suite products, you need to understand your options before installing the product.

This chapter contains the following sections:

- "Understanding Schema Choices" on page 101
- "Understanding Provisioning Tools" on page 106

Understanding Schema Choices

This section describes the schema options that are available and supported with Communications Suite, 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 *Sun Java System Communications Services 6 2005Q4 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.

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 106.

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 Suite 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 Suite 5.x installation that uses Schema 1, and you want to integrate Communications Suite with other Java Enterprise Server products, you should migrate your directory to Schema 2 after you upgrade to Communications Suite 6. Refer to the *Sun Java System Communications Services 6 2005Q4 Schema Migration Guide* for information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2.

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 Delegated Administrator. For more information, see "Understanding Provisioning Tools" on page 106.

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 6 2005Q4 Schema Migration Guide* for more information.

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 6 2005Q4 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.

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 106.

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 Suite 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 Suite 5.x installation that uses Schema 1, and you want to integrate Communications Suite with other Java Enterprise Server products, you should migrate your directory to Schema 2 after you upgrade to Communications Suite 6. Refer to the *Sun Java System Communications Services 6 2005Q4 Schema Migration Guide* for information on how to migrate from LDAP Schema version 1 to LDAP Schema version 2.

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 Delegated Administrator. For more information, see "Understanding Provisioning Tools" on page 106.

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 6 2005Q4 Schema Migration Guide* for more information.

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 101, you should use Table 8–1 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" on page 106
- "LDAP Provisioning Tools for Messaging Server" on page 107
- "Delegated Administrator and Messaging Server" on page 107
- "Comparing Messaging Server Provisioning Tool Options" on page 107

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 Delegated Administrator 6.4. 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 Delegated Administrator 6.4 Administration Guide* for installation details.

Comparing Messaging Server Provisioning Tool Options

Table 8–1 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 1.2 documentation. Describes how to install and administer the Sun ONE Delegated Administrator interface.

TABLE 8–1 Messaging Server Provisioning Mechanisms (Continued)

Supported Provisioning Tool	Provisioning Tool Functionality	Provisioning Tool Limitations	For Further Information
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 1.2 documentation. Provides syntax and usage for Sun ONE Delegated Administrator command-line utilities.
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 *iPlanet Messaging Server 5.2 Provisioning Guide* and the *iPlanet Messaging and Collaboration Schema Reference*. 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.
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 interface.	Read the *Sun Java System Delegated Administrator 6.4 Administration 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 104, you should use Table 8–2 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" on page 107
- "Delegated Administrator and Calendar Server" on page 110
- "Comparing Messaging Server Provisioning Tool Options" on page 107

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 Delegated Administrator 6.4. 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.

See the *Sun Java System Delegated Administrator 6.4 Administration Guide* for installation details.

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 *iPlanet Messaging Server 5.2 Provisioning Guide* and the *iPlanet Messaging and Collaboration Schema Reference*. 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.
Delegated Administrator Uses: Schema 2	Provides graphical and command-line interfaces for administrators to manage users, groups, domains, and resources. Compatible with other Java Enterprise System products.	■ Not backwardly compatible with Sun Schema 1. ■ Sun Java System Access Manager must be installed to enable this interface.	Read the *Sun Java System Delegated Administrator 6.4 Administration Guide*. Provides syntax and usage for the command-line utility.

PART II

Deploying Messaging Server

This part contains the following chapters:

- Chapter 9, “Introduction to Messaging Server Software,”
- Chapter 10, “Planning a Messaging Server Sizing Strategy,”
- Chapter 11, “Developing a Messaging Server Architecture,”
- Chapter 12, “Designing a Messaging Server Topology,”
- Chapter 13, “Planning Messaging Server Security,”
- Chapter 14, “Planning a Messaging Server Anti-Spam and Anti-Virus Strategy,”

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 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?" on page 115
- "Messaging Server Support for Standards and Functionality" on page 116
- "Messaging Server Software Architecture" on page 119

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, "Supported Standards," in *Sun Java System Messaging Server 6.3 Administration Reference*.

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 Appendix A, "Messaging Server Pre-Installation Considerations and Procedures," in *Sun Java Communications Suite 5 Installation Guide* for more information.

Messaging Server provisioning for Schema 2 is done using Delegated Administrator, as documented in the *Sun Java System Delegated Administrator 6.4 Administration Guide*. Delegated Administrator provides a graphical user interface and a set of command-line utilities for managing the users, groups, domains, resources, and service packages within an organization.

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 5.2 Provisioning Guide* - Describes how to create domain, user, group, or administrator entries using LDAP
- *iPlanet Messaging and Collaboration Schema Reference* - Describes Schema 1 for Communications Suite
- *iPlanet Messaging Server 5.2 Reference Manual* - Describes the iPlanet Delegated Administrator command-line utilities for managing users, groups, and domains
- 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 Delegated Administrator 6.4 Administration Guide*, is the recommended mechanism for provisioning Communications Suite users.

Support for Unified Messaging

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

Support for Webmail

Messaging Server currently ships with two native 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/app/docs/coll/1312.1

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 email encryption using S/MIME
- 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
- Support for milter-based email filtering

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 complete set of command-line utilities.

See the *Sun Java System Messaging Server 6.3 Administration Guide* for more information.

Messaging Server Software Architecture

Figure 9–1 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.

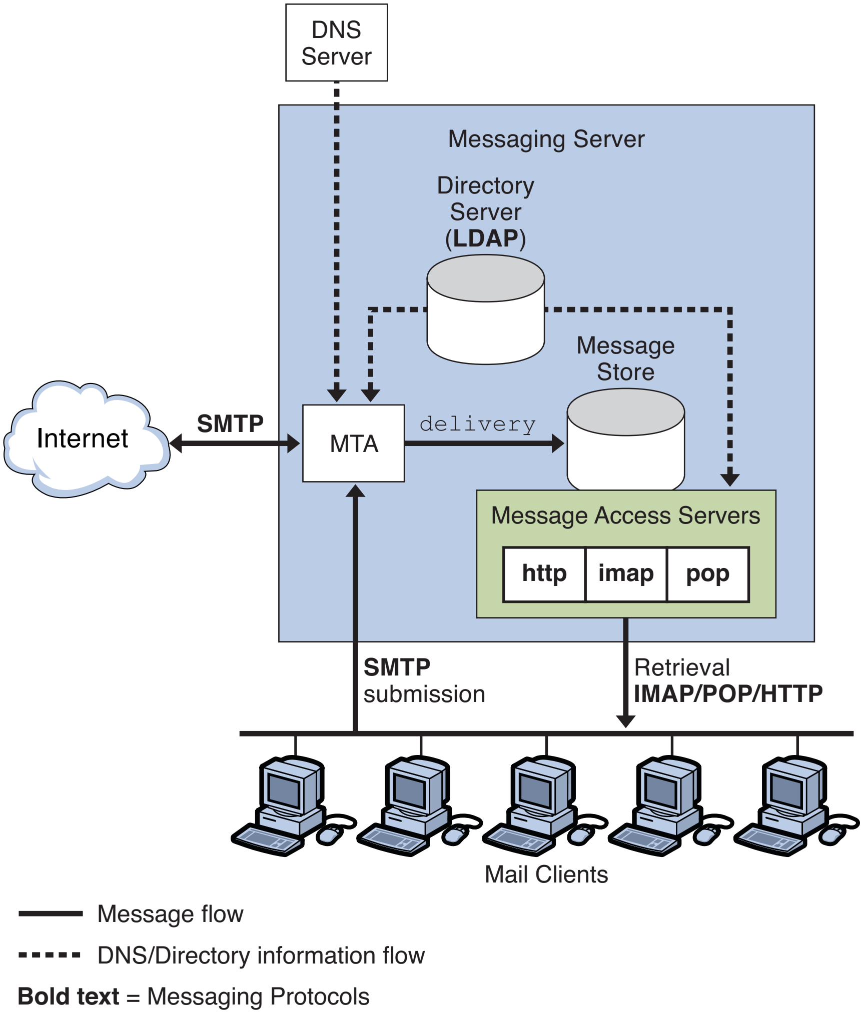

FIGURE 9–1 Standalone Messaging Server, Simplified Components View

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, 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), which was the case in previous versions of the Messaging Server, local messages are usually delivered to the more optimized Messaging Server Message Store. Messages are then retrieved by IMAP4, POP3, or HTTP mail client programs.

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 Delegated Administrator utility or by modifying the directory using LDAP.

Messaging Server provides a set of command-line administrative interfacs 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.

Figure 9–2 illustrates the process. You can configure channels individually and direct mail to specific channels based on the address.

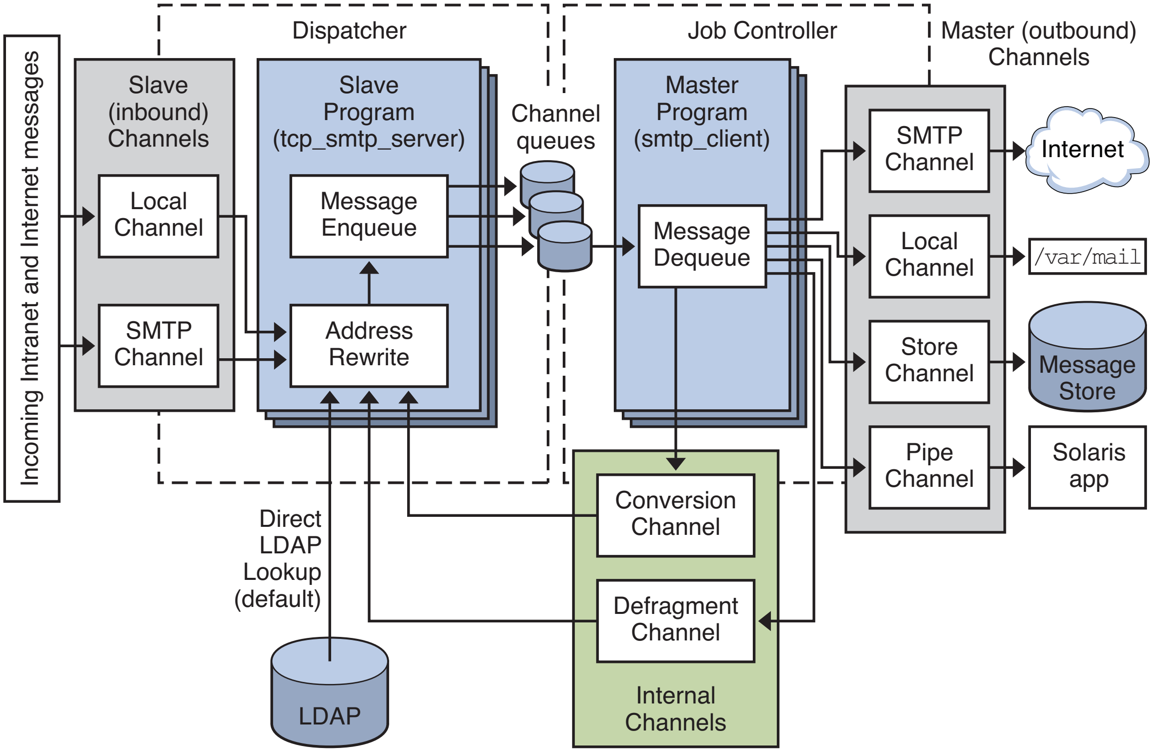

FIGURE 9–2 Channel Architecture

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 Chapter 8, “MTA Concepts,” in *Sun Java System Messaging Server 6.3 Administration Guide* for more information on MTA concepts.

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 6.3 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 Chapter 11, "Configuring Rewrite Rules," in *Sun Java System Messaging Server 6.3 Administration Guide*.

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 server 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 Chapter 16, "LMTP Delivery," in *Sun Java System Messaging Server 6.3 Administration Guide* for more information.

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 Chapter 20, "Managing the Message Store," in *Sun Java System Messaging Server 6.3 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 Chapter 20, "Managing the Message Store," in *Sun Java System Messaging Server 6.3 Administration Guide*.

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/app/docs/coll/1316.1`

Provisioning Messaging Users

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

Message Archiving Using the AXS-One System

Messaging Server supports archiving through the AXS-One message archiving system. Whether you require archiving for regulatory, compliance or legislative purposes, or you wish to manage the growth of your message store or reduce the storage costs, the AXS-One messaging archiving system can help them to achieve both, individually, or simultaneously.

For more information on how to connect the AXS-One archiving system to Sun Java System Messaging Server, see *Message Archiving Using the Sun Compliance and Content Management Solution*.

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" on page 130
- "Using a Messaging Server Load Simulator" on page 136
- "Assessing Your Messaging Server System Performance" on page 136
- "Developing Messaging Server Architectural Strategies" on page 141

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" on page 130
- "Creating Your Messaging Usage Profile" on page 130
- "Defining Your Messaging User Base" on page 134

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:

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 *Sun Java Communications Suite 5 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:

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.

Connection	Description
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 6.3 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 MMP proxy servers and separate MTA layers.
4. What are your response time requirements? What are your throughput requirements?
5. What is your specific criteria for resource utilization? Can your CPUs be 80 percent busy on average? Or only at peak?
6. Will you have messaging servers at different geographic locations? Do you expect users' mail to be located geographically?
7. Do you have archiving requirements for keeping mail messages for a certain length of time?
8. Do you have legal requirements to log all messages? Do you need to keep a copy of every message sent and received?

Defining Your Messaging User Base

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

- "Lightweight POP" on page 134
- "Heavyweight POP" on page 135
- "Lightweight IMAP" on page 135
- "Mediumweight IMAP" on page 135
- "Mediumweight Messenger Express/Communications Express" on page 135

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 136).

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" on page 136.

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 134) and your messaging usage profile (described in "Creating Your Messaging Usage Profile" on page 130) 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/O operations per second) not bandwidth. You need to measure the number of unique disk transactions the system can handle with a very low response time (less than 10 milliseconds).

Messaging Server Disk Capacity

When planning server system disk space, you need to be sure to include space for operating environment software, Messaging Server software, and message content and tracking. 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.

Disk Sizing for MTA Message Queues

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

Message Queue Performance

Sizing the MTA Queue disks are an important step for improving MTA performance. The MTA's performance is directly tied to disk I/O first above any other system resource. This means that you should plan on disk volume that consists of multiple disk spindles, which are concatenated and stripped by using a disk RAID system.

End users are quickly affected by the MTA performance. As users press the SEND button on their email client, the MTA will not fully accept receipt of the message until the message has been committed to the Message Queue. Therefore, improved performance on the Message Queue results in better response times for the end-user experience.

Message Queue Availability

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

Message Queue Available Disk Sizing

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

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

The following sections address these issues.

Planning for Network Connectivity Issues

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

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

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

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

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

Tuning MTA for Reattempts of Delivery

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

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

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

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

The following is the default channel block:

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

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

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

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

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

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

Further Readings

Refer to the following documentation for more information.

- "To Set Notification Message Delivery Intervals" in *Sun Java System Messaging Server 6.3 Administration Guide*
- Chapter 12, "Configuring Channel Definitions," in *Sun Java System Messaging Server 6.3 Administration Guide*

Messaging Server Network Throughput

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

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

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

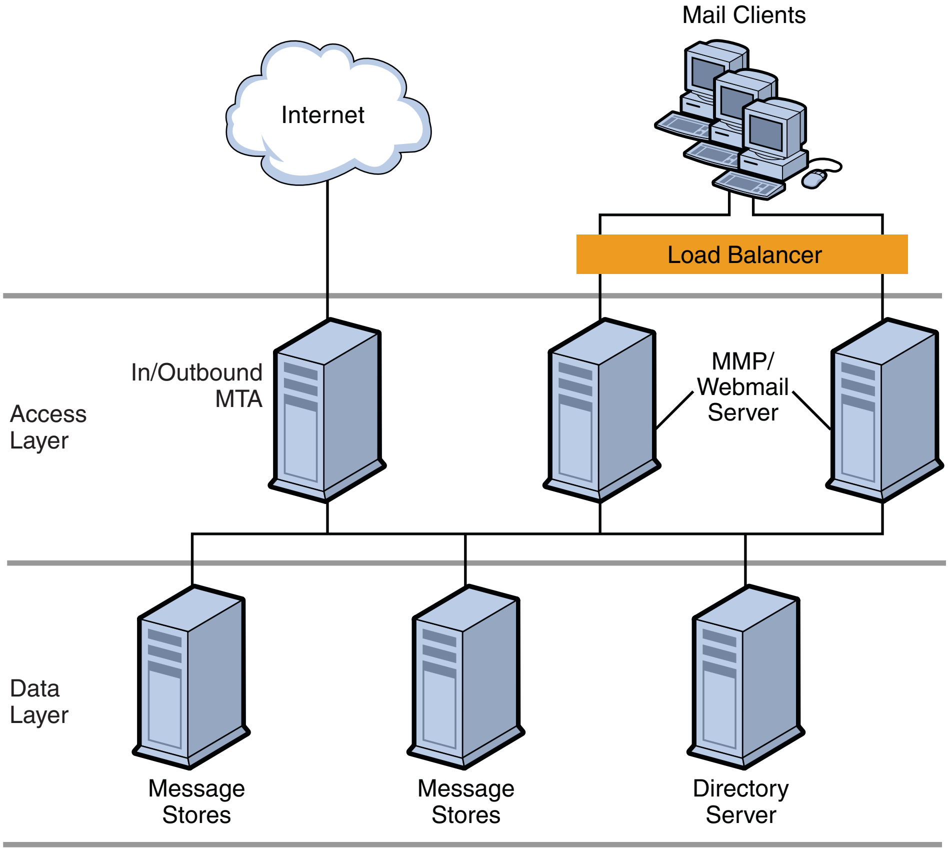

FIGURE 10–1 Simplified Messaging Server Two-tiered Architecture

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 136.**

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 136.
- 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 MMP

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

In addition, you must:

1 **Add CPU or a hardware accelerator for SSL.**

2 **Add more disks for an SMTP proxy.**

3 **Account for denial of service.**

4 **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 shows a single-tiered architecture.

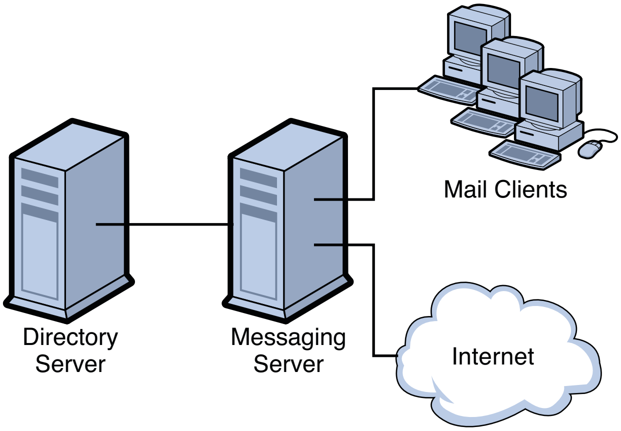

FIGURE 10–2 Simplified Messaging Server Single-tiered Architecture

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 141.**

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" on page 147
- "Understanding Horizontal and Vertical Scalability in Messaging Server" on page 151
- "Planning for a Highly Available Messaging Server Deployment" on page 155
- "Performance Considerations for a Messaging Server Architecture" on page 156

Understanding the Two-tiered Messaging Architecture

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

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

Figure 11–1 shows an example two-tiered architecture.

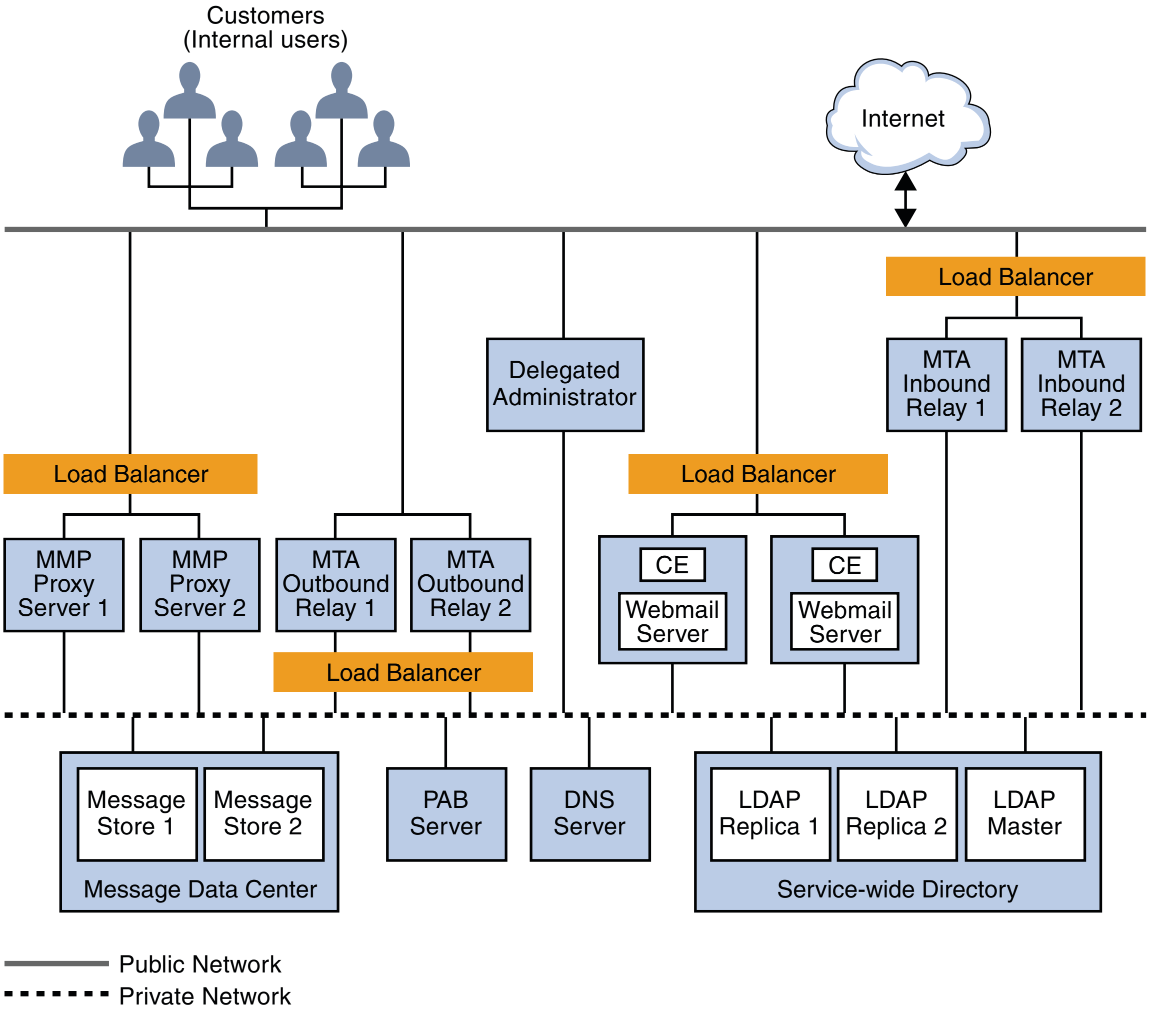

FIGURE 11–1 Two-Tiered Messaging Server Architecture

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 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 (mshttpd) servers. Figure 11–1 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 shows load balancers for the MMPs and the MTAs. Load balancers are not part of Sun Java Communications Suite. You cannot use load balancers on the Message Store or directory masters. You use them for connections to MMPs, Communications Express, MTAs, directory consumers, and with the 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 154 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.

Webmail Server or `mshttpd` daemon. Provides email services to the Messenger Express and Communications Express clients by using HTTP. In previous versions of Messaging Server, the Webmail Server accessed the Message Store directly. Now, the Webmail Server accesses the Message Store through the IMAP server. Such an architecture enables Messenger Express and Communications Express clients to access shared folders that are located in different back-end Message Stores. Additionally, there is no longer a requirement to install the Webmail Server on each back-end server. The Webmail Server can act as a front-end server performing the multiplexing capabilities previously performed by Messenger Express Multiplexor (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/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 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 growth in user population. Scalability also determines how well your system can adapt to significant changes in user behavior, for example, when a large percentage of your users want to enable SSL within a month.

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

- "Planning for Horizontal Scalability" on page 152
- "Planning for Vertical Scalability" on page 155

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" on page 152
- "Spreading Your Messaging Resources Across Redundant Components" on page 153

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.

Figure 11–2 shows a sample deployment where users are spread across multiple back-end servers and multiplexors enabled to handle incoming client connections.

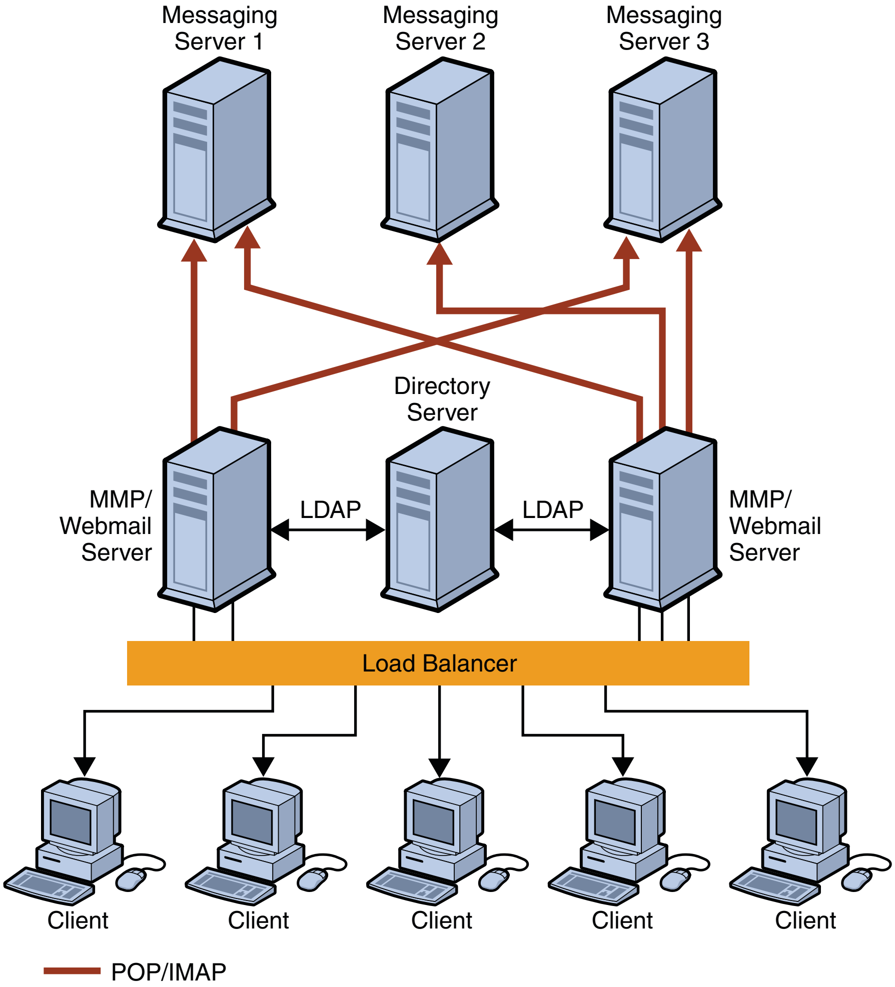

FIGURE 11–2 Spreading Your User Base Across Multiple Servers

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

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" on page 154, and "Inbound and Outbound MTAs" on page 154 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 141.
- Test the load of a prototype of your system

  See "Using a Messaging Server Load Simulator" on page 136.
- 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" on page 156
- "MTA Performance Considerations" on page 162
- "MMP Performance Considerations" on page 163
- "Messaging Server and Directory Server Performance Consideration" on page 164

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. Use of S/MIME
5. Login rate (POP/IMAP/HTTP)
6. Transaction rate for IMAP and HTTP
7. Concurrent number of connections for the various protocols
8. Network I/O
9. 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. If you require a deployment that is scalable, responsive, and resilient to variations in load, provide each of those directories with sufficient I/O bandwidth. When you provide separate file systems for these directories, each composed of multiple drives, you can more readily diagnose I/O bottlenecks and problems. Also, you can isolate the effect of storage failures and simplify the resulting recovery operations. In addition, place a seventh directory for DB snapshots on a file system separate from the active DB to preserve it in the event of a storage failure of the active DB file system.

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 6.3 Administration Reference*. Default location: `/var/opt/SUNWmsgsr/queue`
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 Logical Unit Number (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.

▼ To Adjust the Mailbox Database Cache Size

1 **Become the `mailsrv` user (or whatever user you set `mailsrv` to).**

Using `root` or any other user for this task can cause problems with the database.

2 **Set the `LD_LIBRARY_PATH` to `/opt/SUNWmsgsr/lib`.**

3 **Set the size of the cache with the `configutil` parameter `store.dbcachesize`.**

▼ To Monitor the Mailbox Database Cache Size

Beginning with the Messaging Server 6.3 release, use the `imcheck` command to measure the cache hit rate. Prior to the Messaging Server 6.3 release, use the database command `db_stat`.

1 **Display the cache hit rate.**

Messaging Server 7.0: Run the `imcheck -s mpool` command.

Messaging Server 6.3: Run the `imcheck -s` command.

Messaging Server 6.2: Run the `db_stat` command as follows. In this example, the `configutil` parameter `store.dbtmpdir` has redefined the location of the mailbox database to `/tmp`, that is, `/tmp/mboxlist`.

```
# /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%).
```

2 **Examine the cache hit rate.**

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

3 **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 (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.

MMP and Webmail Server

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

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 F, “Consolidating ACIs for Directory Server Performance,” in *Sun Java System Delegated Administrator 6.4 Administration Guide*.

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" on page 165
- "Designing a Messaging Topology" on page 166
- "Understanding Messaging Topology Elements" on page 173
- "Creating a Messaging Topology Example" on page 177

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" on page 166

  Consolidates most or all major system components and messaging servers at a single location.
- "Distributed Topology" on page 168

  Spreads most or all system components and messaging servers across multiple sites.
- "Hybrid Topology" on page 170

  Consolidates some system components and distributes other components across multiple locations.
- "Service Provider Topology" on page 172

  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 shows a central topology.

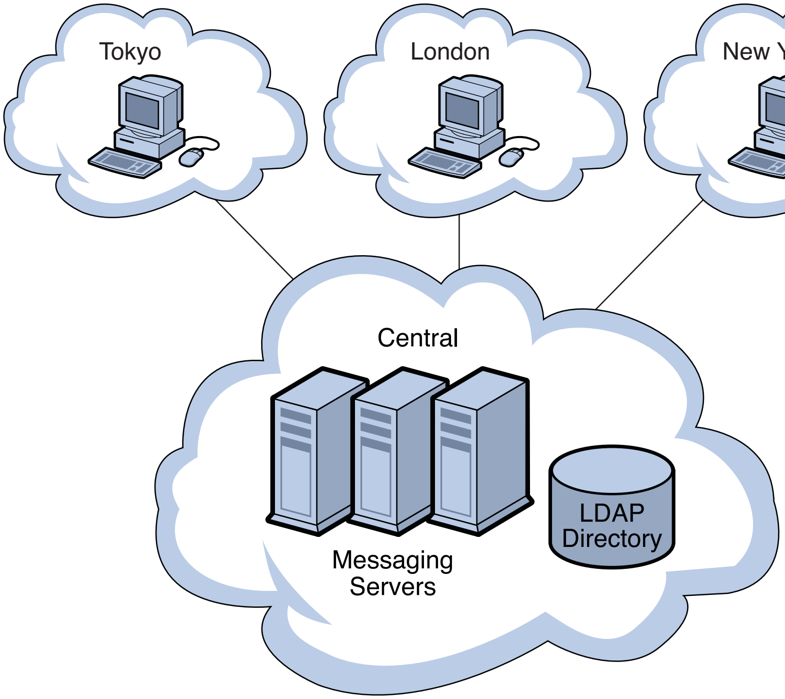

FIGURE 12–1 Central Topology

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, 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 shows a distributed topology.

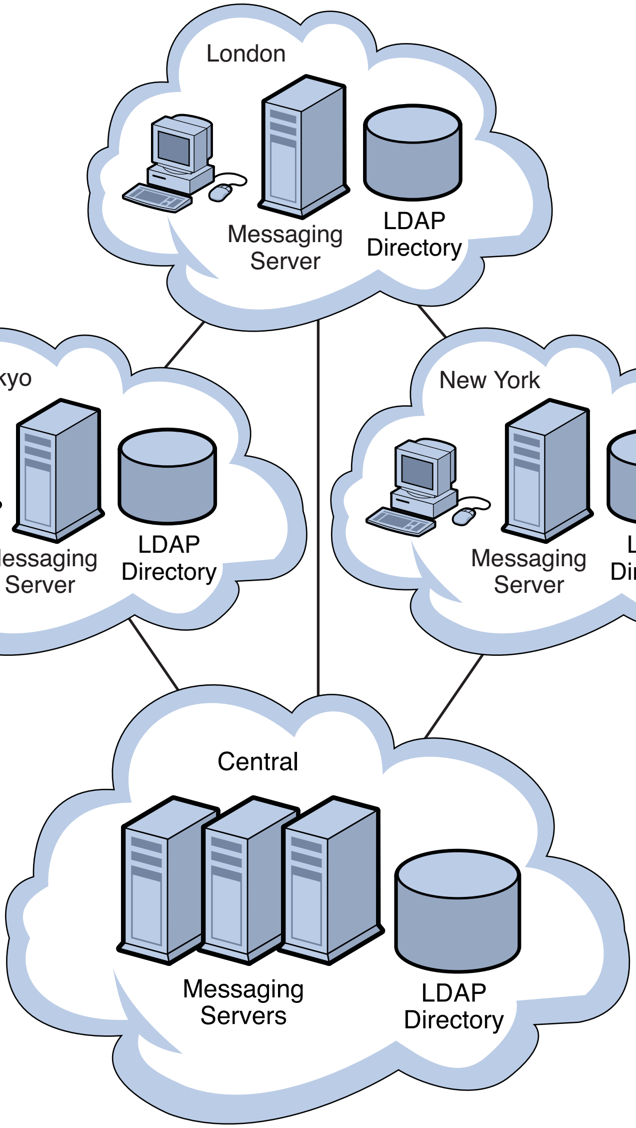

FIGURE 12–2 Distributed Topology

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 shows a hybrid topology.

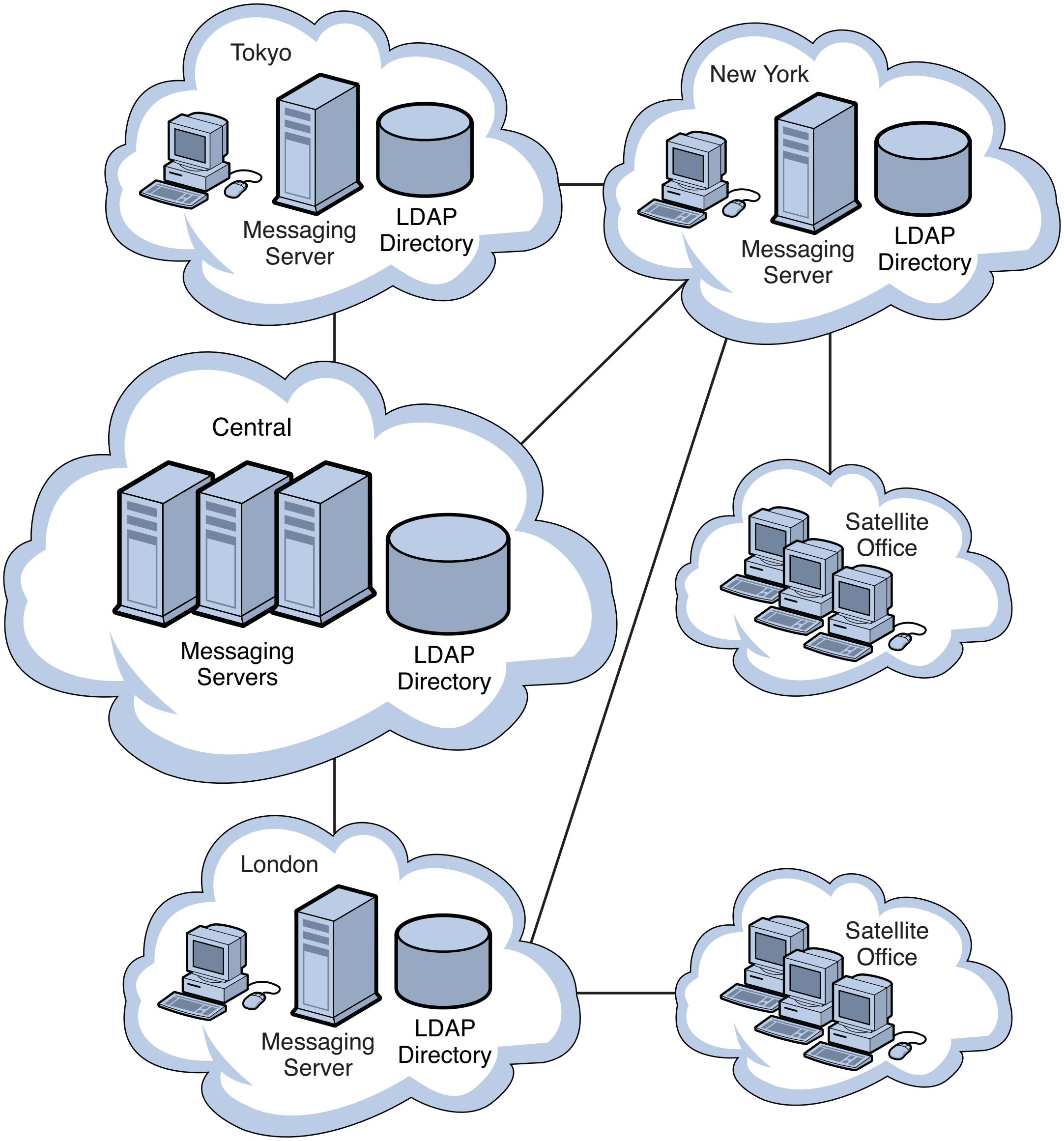

FIGURE 12–3 Hybrid Topology

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 shows a service provider topology.

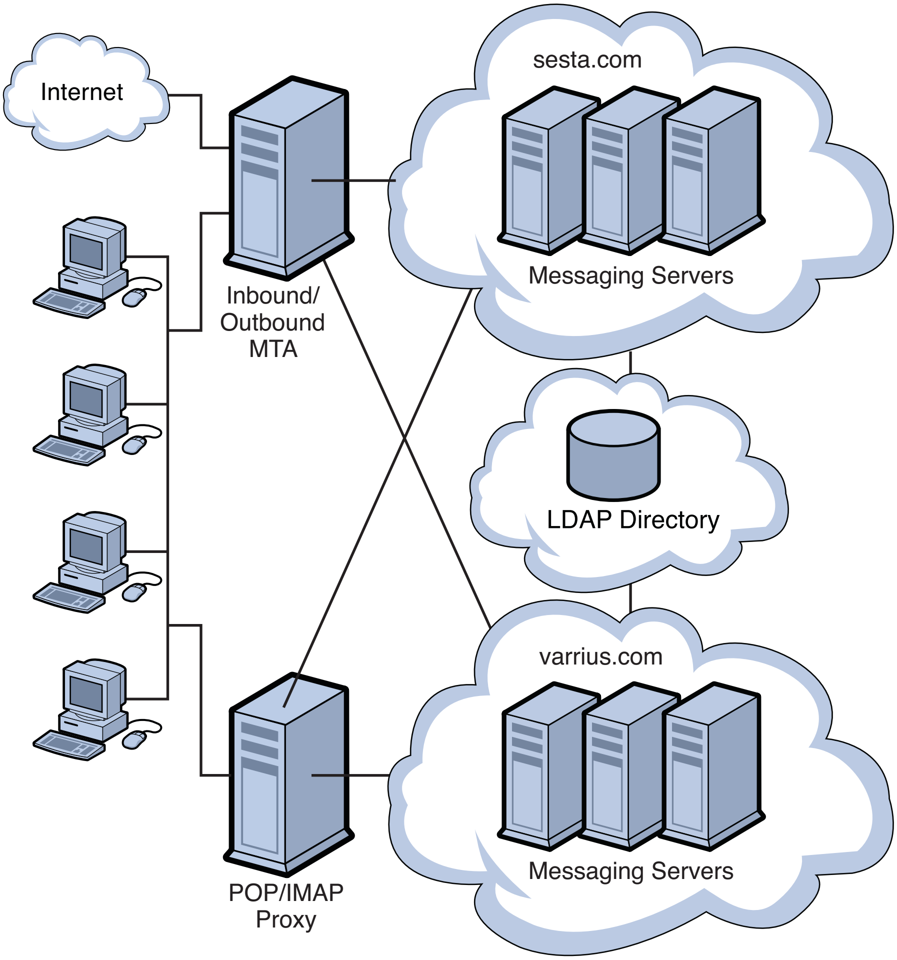

FIGURE 12–4 Service Provider Topology

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" on page 173
- "Using MTAs to Protect Your Messaging System" on page 174
- "Using MMPs" on page 175
- "Using Gateways" on page 177

Messaging Topology Components

In "Designing a Messaging Topology" on page 166, 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 illustrates the idea.

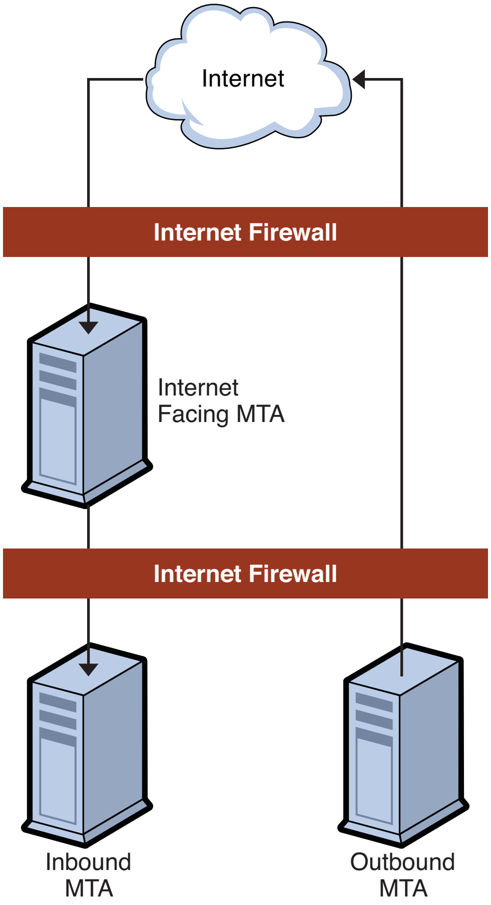

FIGURE 12–5 MTAs in Messaging Topology

Using MMPs

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

When such a client connects and authenticates, the MMP looks up the user information in the directory to determine where the user's messages are held. The MMP then connects the client to that specific server. The following figure shows how the MMP acts as a proxy for IMAP4 and POP3 connections to Messaging servers. Figure 12–6 shows how multiplexors function in a Messaging Server environment.

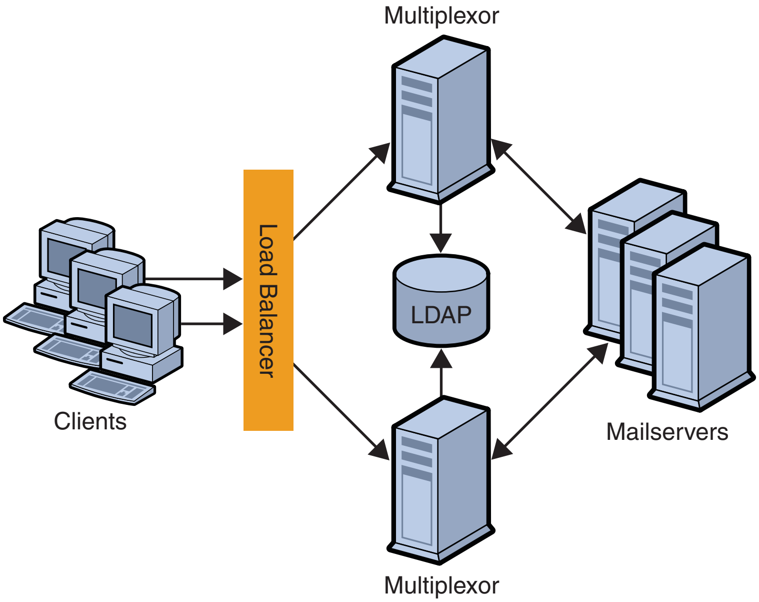

FIGURE 12–6 MMP Overview

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

Using the MMP SMTP Proxy

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

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

Only use the MMP SMTP Proxy in the following situations:

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

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

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

Using Gateways

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

Creating a Messaging Topology Example

Once you have a basic understanding of your topological needs, your strategy, and the topology elements, you can create your messaging topology. To illustrate how to create a messaging topology, this section uses the example of the 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 Suite 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 166. 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 shows that the Siroe Corporation has chosen a hybrid topology.

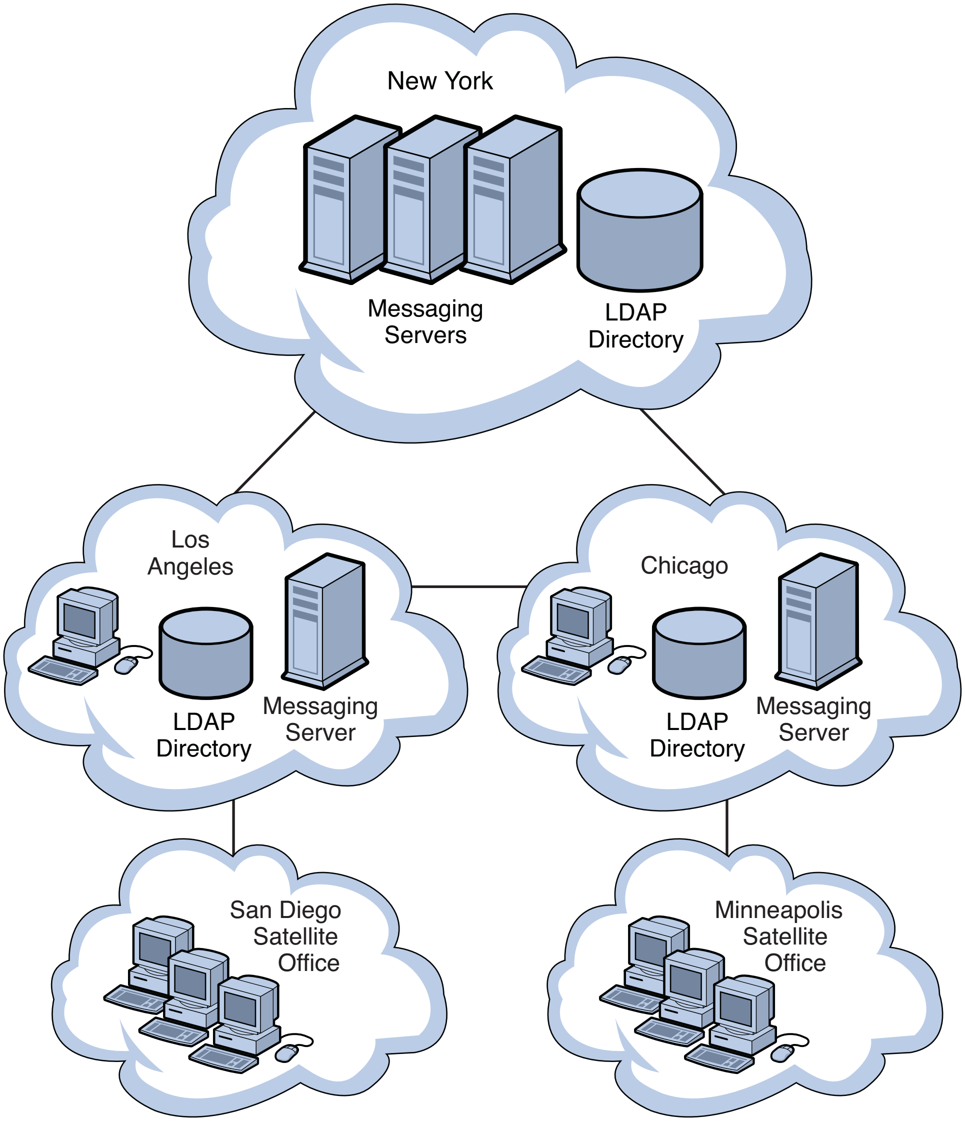

FIGURE 12–7 Hybrid Topology for the Siroe Corporation

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 173. The following figure illustrates the topology elements in the Chicago and Minneapolis offices.

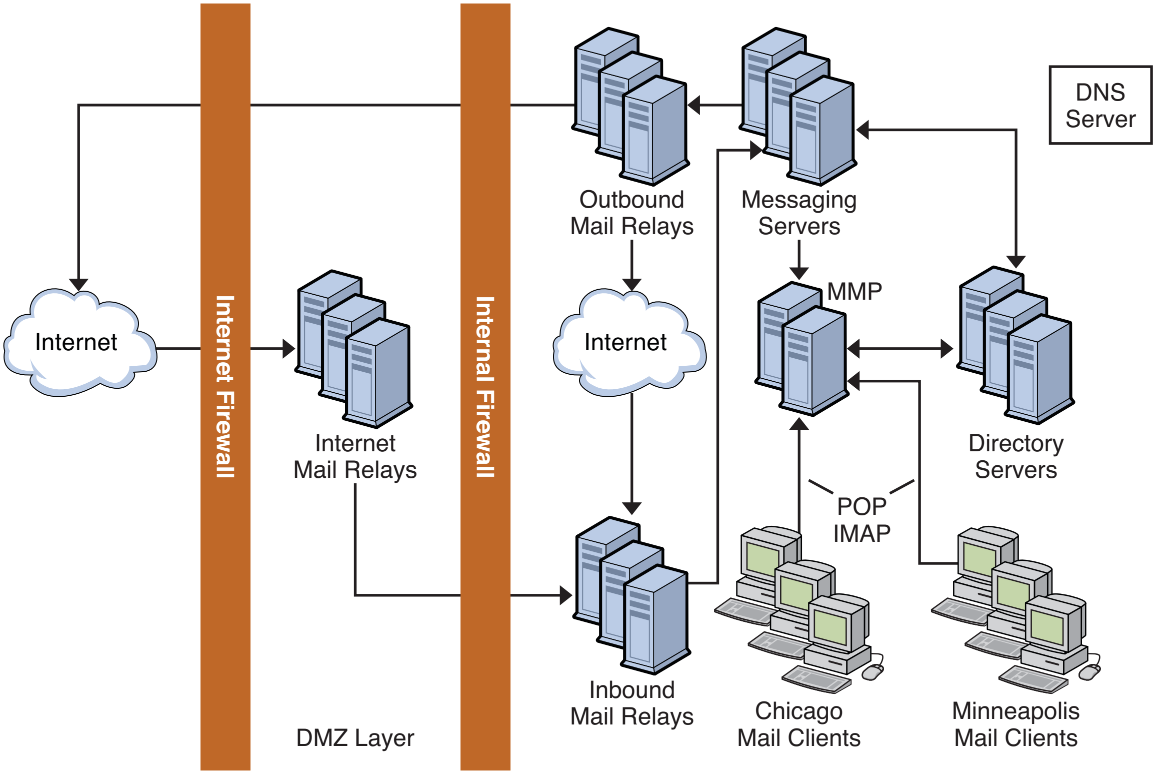

FIGURE 12–8 Topological Elements in the Siroe Messaging Deployment for Chicago and Minneapolis

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" on page 183
- "Planning Messaging User Authentication" on page 192
- "Planning Message Encryption Strategies" on page 196

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" on page 184
- "Conversion Channels and Third Party Filtering Tools" on page 188
- "RBL Checking" on page 189
- "Client Access Filters" on page 189
- "Monitoring Your Security Strategy" on page 190

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 6.3 Administration Guide* for more information.

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`.
`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.

Figure 13–1 illustrates where mapping tables are activated in the mail acceptance process.

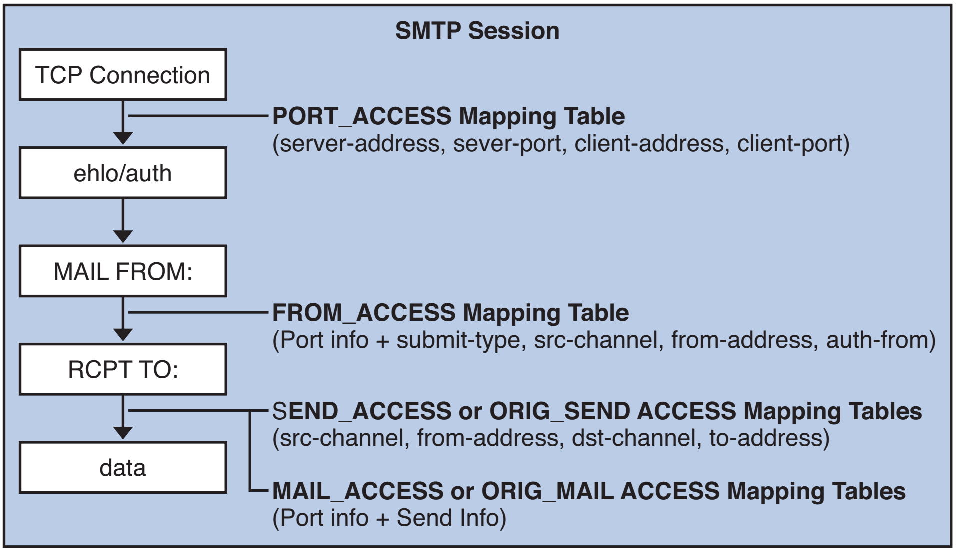
SMTP Session
TCP Connection
ehlo/auth
MAIL FROM:
RCPT TO:
data
PORT_ACCESS Mapping Table
(server-address, sever-port, client-address, client-port)
FROM_ACCESS Mapping Table
(Port info + submit-type, src-channel, from-address, auth-from)
SEND_ACCESS or ORIG_SEND ACCESS Mapping Tables
(src-channel, from-address,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

FIGURE 13–1 Mapping Tables and the Mail Acceptance Process

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

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

Configuring Anti-Relaying with Mapping Tables

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

By default, Messaging Server prevents all SMTP relaying activity, including relaying by local POP and IMAP mail clients. If clients do not authenticate by using SMTP AUTH, as described in “Enabling Authenticated SMTP” on page 194, 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 6.3 Administration Guide*.

Using Mailbox Filters

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

You can create channel-level filters and MTA-wide filters to prevent the delivery of unwanted mail. You can also create filter templates and make them available to end users by using Messenger Express. End users use the templates to build personal mailbox filters to prevent delivery of unwanted mail message to their mailboxes. The server applies filters in the following priority. See the *Sun Java System Messaging Server 6.3 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 Chapter 18, "Mail Filtering and Access Control," in *Sun Java System Messaging Server 6.3 Administration Guide* for specific instructions on creating filters with Sieve.

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

3. MTA-wide filter

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

   To create an MTA-wide filter, you must write the filter using Sieve. See Chapter 18, "Mail Filtering and Access Control," in *Sun Java System Messaging Server 6.3 Administration Guide* for specific instructions on creating filters with Sieve.

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

Conversion Channels and Third Party Filtering Tools

The conversion channel performs body-part-by-body-part conversions on messages through the MTA. This processing can be done by any site-supplied programs or command procedures. The conversion channel can do such things such as convert text or images from one format to another, scan for viruses, translate languages, and so forth. Various message types of the MTA traffic are selected for conversion, and specific processes and programs can be specified for each type of message body part. If you are looking to use the conversion channel with a virus scanning program, you can either disinfect, hold, or reject messages. A special conversion channel configuration is consulted to choose an appropriate conversion for each body part. For more information, see Chapter 13, "Using Predefined Channels," in *Sun Java System Messaging Server 6.3 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 6.3 Administration Reference*.

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

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

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

For more information on configuring Brightmail and SpamAssassin for Messaging Server, see Chapter 14, "Integrating Spam and Virus Filtering Programs Into Messaging Server," in *Sun Java System Messaging Server 6.3 Administration Guide*.

RBL Checking

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

You can configure your 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 Chapter 18, "Mail Filtering and Access Control," in *Sun Java System Messaging Server 6.3 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 Chapter 23, "Configuring Security and Access Control," in *Sun Java System Messaging Server 6.3 Administration Guide*.

Monitoring Your Security Strategy

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

Protecting the Message Store

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

To secure the Message Store, restrict access to the machine where the store is installed. You can enable CRAM-MD5 or Digest-MD5 passwords instead of using unencrypted, plaintext passwords. For more information on passwords, see "Planning Messaging User Authentication" on page 192.

Not only should you create password authentication to the store machine, you might also use tools like VPN access, `ssh`, or `pam`, which list 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 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 Chapter 20, "Managing the Message Store," in *Sun Java System Messaging Server 6.3 Administration Guide*.

Protecting MMPs

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 194.

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. Like the MMP, the Webmail Server supports both unencrypted and encrypted (SSL) communication with mail clients.

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" on page 192
- "Authentication with Simple Authentication and Security Layer (SASL)" on page 193
- "Enabling Authenticated SMTP" on page 194
- "Certificate-based Authentication with Secure Sockets Layer (SSL)" on page 194

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. See the Directory Server documentation to understand directory server password policies:

`http://docs.sun.com/coll/1316.1`

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 6.3 Administration Reference*. The `RestrictPlainPasswords` option provides the equivalent function for the MMP

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 Table 13–3:

TABLE 13–3 SASL Authentication User Access Protocols Support Matrix

	Plain	Login	CRAM-MD5	Digest-MD5	Certificate	APOP
SMTP AUTH	Yes	Yes	Yes	Deprecated	Yes	-
POP	Yes	-	Yes	Deprecated	Yes	Yes
IMAP	Yes	-	Yes	Deprecated	Yes	-
HTTP (Messenger Express)	Yes	-	-	-	Yes	-

Note –

- When using CRAM-MD5, passwords must be stored in plain text format in the LDAP directory server.
- Digest-MD5 was never supported by the MMP and its use is deprecated in other cases.
- To use APOP, CRAM-MD5 or DIGEST-MD5, 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 196.) 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 194) to allow only authenticated users 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 Chapter 23, “Configuring Security and Access Control,” in *Sun Java System Messaging Server 6.3 Administration Guide*.

Enabling Authenticated SMTP

By default, the standard SMTP port (25) is for mail transfer only. Mail relay for submissions from external networks is disabled and authentication is disabled. By default, the standard SMTP submit port (587) is for mail submission and requires authenticated SMTP. As many mail user agents still use port 25 for submission by default it might be useful to enable SMTP authentication on port 25 for those clients.

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)" on page 194.)

With authenticated SMTP, the client can indicate an authentication mechanism to the server and perform an authentication protocol exchange.

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 6.3 Administration Guide*.

Certificate-based Authentication with Secure Sockets Layer (SSL)

Messaging Server uses the SSL protocol for encrypted communications and for certificate-based authentication of clients and servers. This section describes certificate-based SSL authentication. For information on SSL Encryptions, see "Encryption with SSL" on page 196.

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

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

Table 13–4 describes the SSL authentication support with each client access protocol. This table shows whether a secure session (startTLS) could be started up over a insecure channel and whether a separate secure channel (SSL on Separate Port) is provided.

TABLE 13–4 SSL Authentication Support Matrix

Protocol	startTLS	SSL on Separate Port
SMTP	Yes	Yes
POP	No	Yes
IMAP	Yes	Yes
POP over MMP	No	Yes
IMAP over MMP	Yes	Yes
SMTP over MMP	Yes	Yes
Webmail	No	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 `startTLS` command. The SMTP, POP, and IMAP servers can also be configured to use SSL on an alternate port, for clients which do not implement `startTLS`.

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.

Some protocols require use of the SASL EXTERNAL mechanism in conjunction with the SSL client certificate to move from un-authenticated to authenticated state.

For more information on SSL, see Chapter 23, "Configuring Security and Access Control," in *Sun Java System Messaging Server 6.3 Administration Guide*.

Planning Message Encryption Strategies

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

- "Encryption with SSL" on page 196
- "Signed and Encrypted S/MIME" on page 197

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 "Requirements for Using S/MIME with Communications Express Mail" in *Sun Java Communications Suite 5 Installation Guide*.

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. Sun UltraSPARC IV computers have built-in CPU support for SSL encryption but it is not enabled by default.

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 Chapter 23, “Configuring Security and Access Control,” in *Sun Java System Messaging Server 6.3 Administration Guide*.

Signed and Encrypted S/MIME

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

With S/MIME, senders can encrypt messages prior to sending them. 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” in *Sun Java Communications Suite 5 Installation Guide* 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” on page 199
- “Anti-Spam and Anti-Virus Considerations” on page 204
- “Common Anti-Spam and Anti-Virus Deployment Scenarios” on page 206
- “Developing an Anti-Spam and Anti-Virus Site Policy” on page 207

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
- **Milter.** Provides a plug-in interface for third-party software to validate, modify, or block messages as they pass through the MTA.

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 6.3 Administration Guide*.

Rejecting Mail

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 184 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) developed by the Internet Engineering Task Force (IETF).

See "Using Mailbox Filters" on page 187 for more information.

You can also implement content-based filtering or 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 204 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's 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 6.3 Administration 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.com`

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 186 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 Messages

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.

Milter

Milter refers to the Sendmail Content Management API and also to software written using this API. Milter provides a plug-in interface for third-party software to validate, modify, or block messages as they pass through the MTA. In sendmail, milter consists of support code in sendmail itself and a separate milter library. Filter authors link their filters against this library to produce a server. Sendmail is then configured to connect to these milter servers. Messaging Server provides a library that emulates the sendmail side of the milter interface. Consequently, milters written for sendmail can also be used with Messaging Server. The milter server can run in a variety of configurations. It can run on a separate system of its own, on the same system as Messaging Server, in a single system deployment, or in a two-tier deployment. Messaging Server also supports connecting to multiple milter servers.

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 in 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.

Security Issues with Anti-Spam and Anti-Virus Deployments

When planning to deploy anti-spam or anti-virus technology, keep in mind that an incorrect deployment can defeat your security measures. Figure 14–1 shows an incorrect deployment of an anti-spam/anti-virus filter solution.

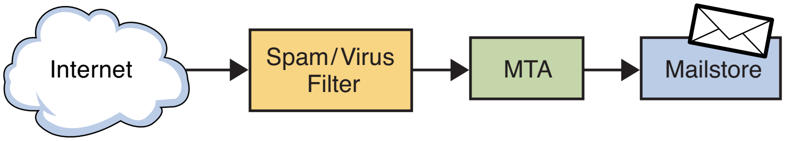

FIGURE 14–1 Incorrect Deployment of Anti-Spam/Virus Solution

Figure 14–2 shows a correct deployment of an anti-spam/virus filter solution.

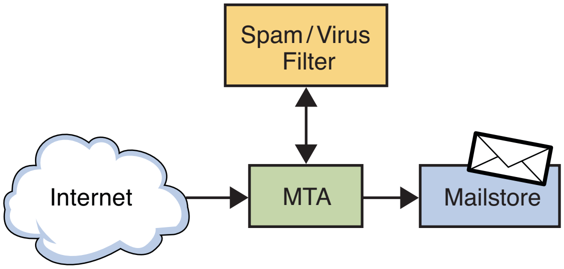

FIGURE 14–2 Correct Deployment of Anti-Spam/Virus Solution

The MTA performs certain functions well, including:

- Rejecting messages as early as possible
- Per-user configuration and policy
- Email security and routing policy
- Mail queue management

The anti-spam/virus filter is good at determining if an email is spam or has a virus, but is generally not nearly as good at doing the things expected of a good MTA. Thus, do not depend on an anti-spam/virus filter to do those things. Your deployment is more "correct" when the anti-spam/virus filter is well integrated with the MTA, which is the case with Messaging Server. Messaging Server spam filter plug-in support provides all the potential reasons to reject a message early and applies all reasons at the same time.

A robust MTA, such as Messaging Server's, contains security features (SSL/TLS, traffic partitioning by IP address, early address rejection to reduce denial-of-service attacks, connection throttling by IP address/domain, and so on), which are defeated when an anti-spam/virus filter is deployed in front. Furthermore, anti-spam/virus filters that communicate by using the SMTP protocol often do not follow the robustness requirements of

SMTP and thus lose email when they shouldn't. A correct deployment should have the anti-spam/virus filter working in conjunction with a robust MTA.

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 6.3 Administration Guide* for more information.

Using Symantec Brightmail

There are several common deployment scenarios for Symantec 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.

PART III

Deploying Calendar Server

This part contains the following chapters:

- Chapter 15, “Introduction to Calendar Server Software,”
- Chapter 16, “Developing a Calendar Server Architecture,”
- Chapter 17, “Planning Calendar Server Security,”
- Chapter 18, “Planning Calendar Server Services,”

CHAPTER 15

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" on page 211
- "Designing Your Calendar Server Deployment" on page 213

Calendar Server Overview

Sun Java System 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 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.

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, Sun Java System Portal Server, and Sun Java System Instant Messaging to provide additional functionality.

Calendar Server provides the following benefits:

- Group scheduling, free-busy lookup, and corporate directory lookup
- Resource scheduling for conference rooms, projectors, and other resources
- 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
- Support for the Connector to Microsoft Outlook
- Support for hosted domains including command-line and GUI tools to manage these 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
- Automatic email 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
- Support for daily, weekly, monthly, and yearly views of a calendar
- 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
- Integrated personal address book and email functionality

- Support for layered calendar views, enabling users to combine two or more calendars into a single view for improved communications and productivity
- Support for attachments in events and tasks
- Support for LDAP groups including static, nested, and dynamic groups

For more information on Calendar Server concepts, see the *Sun Java System Calendar Server 6.3 Administration Guide*.

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.

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 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.
- **High availability (HA) configuration.** Integration with 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 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? Communications Express? Microsoft Outlook? A combination of both 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 an earlier version of Calendar Server?
- Are your end users planning to use Communications Sync Software?
- 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 do you have for growth? For both front-end and back-end servers?

CHAPTER 16

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" on page 217
- "Two-tiered Calendar Server Architecture" on page 219
- "Expected Calendar Server End User Performance" on page 215
- "Two-tiered, Multiple Server Calendar Server Architecture" on page 221

Single-Server Calendar Server Architecture

Figure 16–1 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.

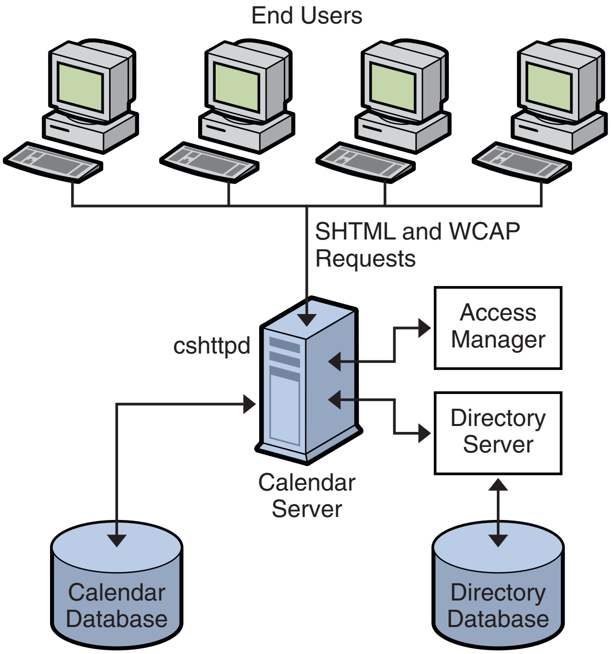

FIGURE 16–1 Single-Server Calendar Server Architecture

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 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 6.3 Administration Guide*.

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 6.3 WCAP Developer's Guide*.

The directory server can run on the same server where Calendar Server is running or on a remote server.

The tool Calendar Server uses for administration of users, groups, resources, domains, and roles for Schema 2 deployments is Delegated Administrator, which must be installed and configured separately. The Delegated Administrator has both a graphical user interface console, and a command line utility (`commadmin`). For information about the Delegated Administrator utility, see the *Sun Java System Delegated Administrator 6.4 Administration Guide*.

For Schema 1 deployments, Calendar Server has its own bundled set of utilities for administration.

You can implement SSO for Communications Suite products using Access Manager, or using trusted circle technology among the Communications Suite products. Users log in to Access Manager and then can access other the servers, as long as all servers are configured properly for SSO. For the trusted circle technology, a user logging into Messaging Server would also have access to Calendar Server and Instant Messaging without having to login again. For more information on SSO trusted circle implementation, see the *Sun Java System Calendar Server 6.3 Administration Guide*.

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. Though not shown in this figure, the front-end hosts need access to the LDAP directory.

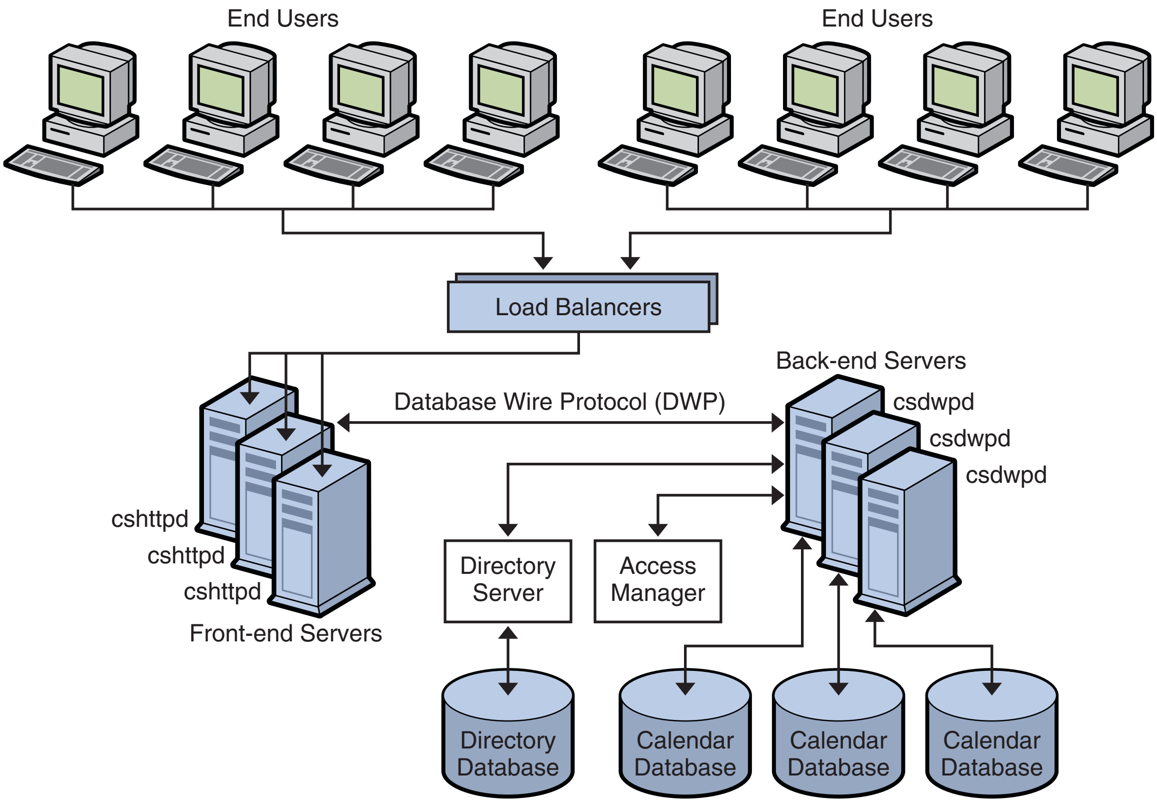

FIGURE 16–2 Two-tiered Calendar Server Architecture

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)
  - GSE database (`csstored`)
- 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 6.3 Administration Guide*.

A scalable Calendar Server configuration requires a directory server to authenticate users and to store user preferences.

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 16–3), 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.

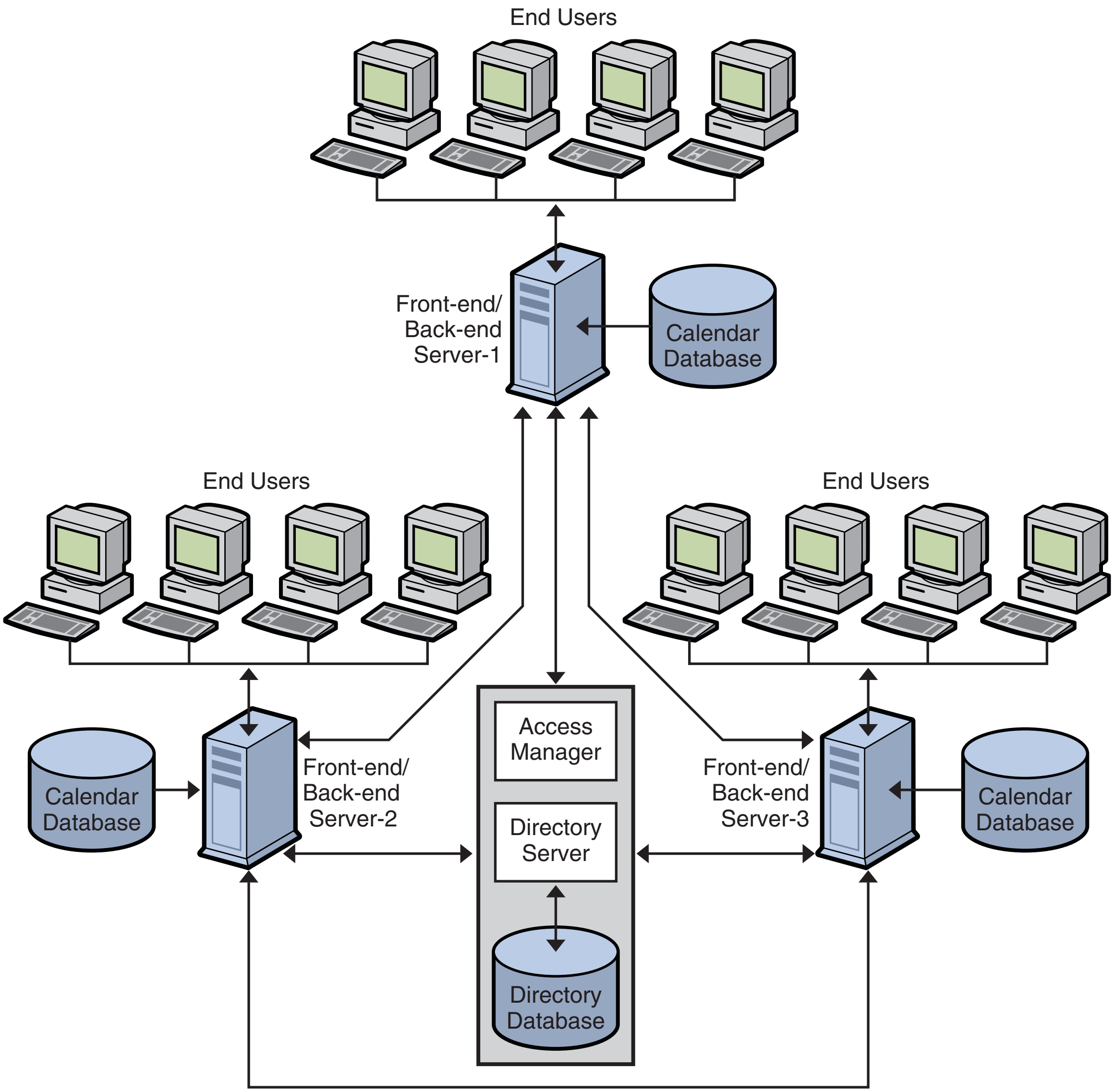

FIGURE 16–3 Two-tiered, Multiple Server Calendar Server Architecture

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), Database Wire Protocol (DWP) Service (`csdwpd` process), and Backup Service (`csstored`).

For a description of Calendar Server services, see the *Sun Java System Calendar Server 6.3 Administration Guide*.

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 6.3 Administration Guide*.

A multiple front-end/back-end server deployment requires a directory server to authenticate users and to store user preferences.

CHAPTER 17

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" on page 225
- "Planning Calendar User Authentication" on page 226

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" on page 226
- "Certificate-based Authentication with Secure Sockets Layer (SSL)" on page 227

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 Directory Server documentation:

`http://docs.sun.com/app/docs/coll/1316.2`

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 6.3 Administration Guide*.

◆ ◆ ◆ CHAPTER 18

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" on page 229
- "Planning for the Calendar Server LDAP Data Cache" on page 231
- "Planning Calendar Server Domains" on page 234

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 todos 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.

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. The calendar databases can be distributed over one or more back-end machines. These back-end machines can be associated with one front-end machine.

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 the number of CPUs as the front-ends.

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. Communications Express deployments with Calendar Server have different characteristics. See the Communications Express documentation for more information.

Note – The only configuration for Calendar front ends that might warrant a true HA solution is one where the Calendar front end is deployed 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. In this case, you 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, deploy the Calendar Server back end with Sun Cluster, as the back end contains persistent data.

Note – In a configuration with both front-end and back-end Calender Server hosts, all hosts must be running 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 LDAP, 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.

The LDAP data cache can ensure that your Calendar Server clients have accurate LDAP data, but can introduce a delay in the availability of committed LDAP data. For example, if your site has deployed a master/slave LDAP configuration, with Calendar Server accessing the master LDAP directory through a slave LDAP directory server, it introduces a delay.

This section covers the following topics:

- "Considerations for " on page 231
- "Using the LDAP Data Cache in a Master/Slave LDAP Configuration" on page 232
- "Resolving the Master-Slave Delay Problem" on page 233
- "Configuring the LDAP Data Cache" on page 234"Configuring the LDAP Data Cache" on page 234

Considerations for

Use these guidelines to determine if your site should configure the LDAP data cache:

- You don't need to configure the LDAP data cache if your Calendar Server accesses the master (or root) LDAP directory directly with no delays in the availability of committed LDAP data.
- If your site has deployed a "Using the LDAP Data Cache in a Master/Slave LDAP Configuration" on page 232 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.

Using the LDAP Data Cache in a 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. In this case, make sure that the `local.ldap.cache.enable` parameter is set to “no” (which is the default).
- If Calendar Server accesses the master LDAP directory server through a slave directory server, LDAP data changes are usually written to the master directory server transparently using an LDAP referral, which in turn replicates the data back to each slave directory server. This introduces a delay in the availability of committed LDAP data. 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.

LDAP Attributes Affected by Delays

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 18–1 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” on page 233.

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 `/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 Solution

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. Since each server has its own LDAP cache, within the delay period, one cache will not know about the user's activities while logged into the another cache.

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 6.3 Administration Guide* for more information.

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.

Planning Calendar Server Domains

For new Calendar Server installations, you must first create a default domain before running the Calendar Server configuration script. To create a default domain, use Delegated Administrator. This means that Delegated Administrator must be installed and configured before Calendar Server is configured.

PART IV

Deploying Instant Messaging

This part contains the following chapters:

- Chapter 19, "Introduction to Instant Messaging Software,"
- Chapter 20, "Planning an Instant Messaging Sizing Strategy,"
- Chapter 21, "Developing an Instant Messaging Architecture,"
- Chapter 22, "Planning Instant Messaging Security,"

CHAPTER 19

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?" on page 237
- "Instant Messaging Core Product Components" on page 238
- "Components Related to Instant Messaging" on page 238
- "Instant Messaging Supported Standards" on page 241
- "Instant Messaging Software Architecture" on page 242
- "Designing Your Instant Messaging Deployment" on page 246

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
- Provides presence updates for users who are online, offline, away, and so on

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 241.
- **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 the Instant Messaging server, but are installed separately. Chapter 21, "Developing an Instant Messaging Architecture," provides more detailed information that illustrates how these servers interact with Instant Messaging.

Web Container

(Required) For any deployment, you need to install a web container, 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 container host.

Instant Messaging requires a web container 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 container is installed. In an Access Manager deployment, you can install these resources on the Access Manager server's host or on a different web container 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 container 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 7.2 Administration Guide*.

Note – Because a proper Directory Server implementation is instrumental to a successful Instant Messaging deployment, also see the Directory Server documentation:

`http://docs.sun.com/app/docs/coll/1316.2`

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 can also be used to archive instant messaging communications. This is in lieu of or in addition to the archive capability provided by Portal Server. 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, service, and policy 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" on page 240
- "Secure Remote Access" on page 241

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 you 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 Portal Server documentation:

http://docs.sun.com/coll/1293.2

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 Messaging 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. You can secure these communications by using TLS encryption.

Instant Messaging uses SMTP to send messages to offline users and for email archiving.

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. Client-to-server communication can take place over HTTP/HTTPS/SOCKS proxy. Also, HTTPBIND is a server component (that uses the HTTP protocol) through which browser-based XMPP clients can communicate with the Instant Messaging server.

Instant Messaging 7.2 is an XMPP 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 Instant Messaging server and AOL, Yahoo, and other instant messaging systems.

Instant Messaging Software Architecture

Figure 19–1 shows the Instant Messaging software architecture.

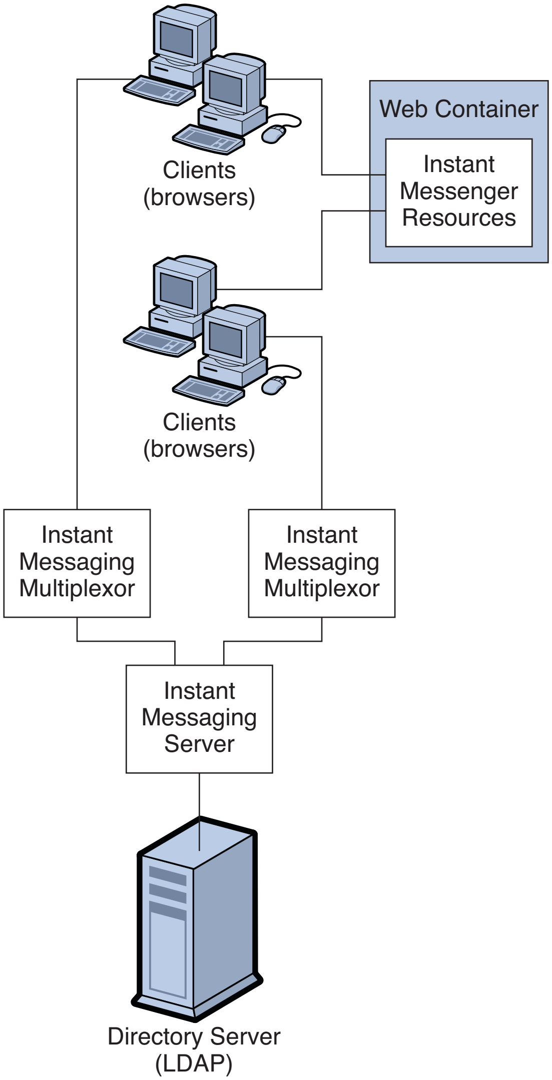

FIGURE 19–1 Instant Messaging Software Architecture

The web server (or an application server with a web service embedded), downloads the Instant Messenger 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 275 for illustrations of this process.)

New users are created by adding user entries to the directory. Entries in the directory can be created through Instant Messaging Server (by enabling new-user registration feature) 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 21, "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" on page 244
- "Instant Messaging Multiplexor" on page 245
- "Instant Messenger Client" on page 245

Instant Messaging Server

The Instant Messaging server handles tasks such as maintaining presence information for each user, 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" on page 245.

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 Directory Server documentation for further information.

`http://docs.sun.com/app/docs/coll/1316.2`

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 261 for an 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.

Although you can configure Instant Messaging without a multiplexor, production deployments should be configured to use the multiplexor. You can install multiple multiplexors based on your deployment requirements. For more information, Chapter 21, "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 7.2 Administration Guide*.

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 7.2 Administration Guide*.

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 Enterprise System Deployment Planning Guide*.

CHAPTER 20

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" on page 249
- "Collecting Instant Messaging Sizing Data" on page 250
- "Using an Instant Messaging Load Simulator" on page 254
- "Understanding Instant Messaging System Performance Guidelines" on page 254
- "Developing Instant Messaging Architectural Strategies" on page 257
- "Example Instant Messaging Resource Requirements" on page 260

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" on page 250
- "Creating Your Instant Messaging Usage Profile" on page 250
- "Defining Your Instant Messaging User Base or Site Profile" on page 253

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.

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.

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 7.2 Administration Guide*.

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 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 7.2 Administration Guide* for more information.

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" on page 253
- "Heavy Users" on page 253

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 254).

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 253) and your usage profile (described in "Creating Your Instant Messaging Usage Profile" on page 250) 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 250. For information on hosting the server and multiplexor on the same host, see "Developing Instant Messaging Architectural Strategies" on page 257.

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 7.2 Administration Guide*.

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 Table 20–1 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 20–1 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 Table 20–2 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 20–2 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 7.2 Administration Guide* for more detailed information about these parameters.

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 relays messages to the Instant Messaging server. The data layer holds the Instant Messaging server database and Directory servers. Figure 20–1 shows a simplified two-tiered Instant Messaging architecture.

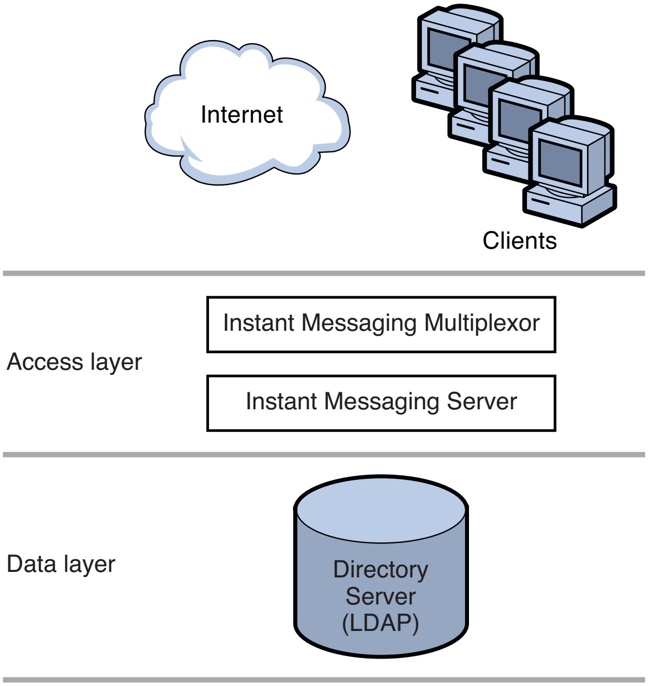

FIGURE 20–1 Simplified Two-tiered Instant Messaging Architecture

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.

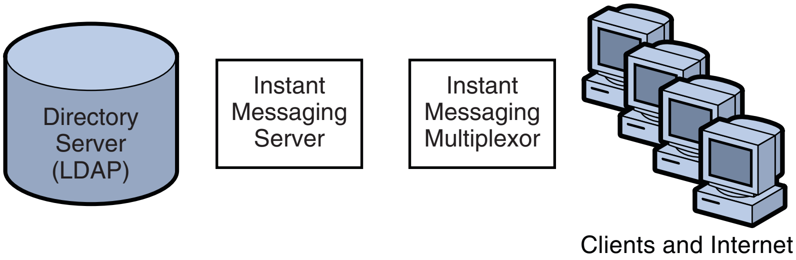

FIGURE 20–2 Simplified One-tiered Instant Messaging Architecture

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" on page 260
- "Large Deployment Sample Resource Requirements Numbers" on page 260

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 hardware 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 21

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 238.

This chapter contains the following sections:

- "Basic Instant Messaging Architecture" on page 261
- "Instant Messaging Email Notification (Calendar Alert) Architecture" on page 265
- "Instant Messaging Access Manager or SSO Architecture" on page 268
- "Instant Messaging Portal-based or Archiving Architecture" on page 271
- "Instant Messaging With All Features Enabled" on page 275
- "Instant Messaging Physical Deployment Examples" on page 275

Note – Currently, all deployment options are available on Solaris OS. A subset of the deployment options is available on the Linux operating system.

Basic Instant Messaging Architecture

Figure 21–1 shows the basic Instant Messaging architecture.

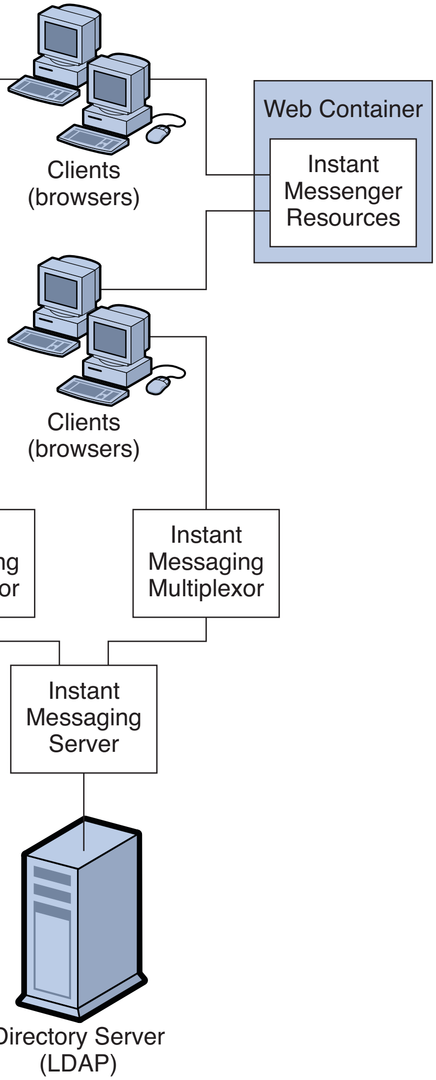

FIGURE 21–1 Basic Instant Messaging Architecture

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 21–2 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.

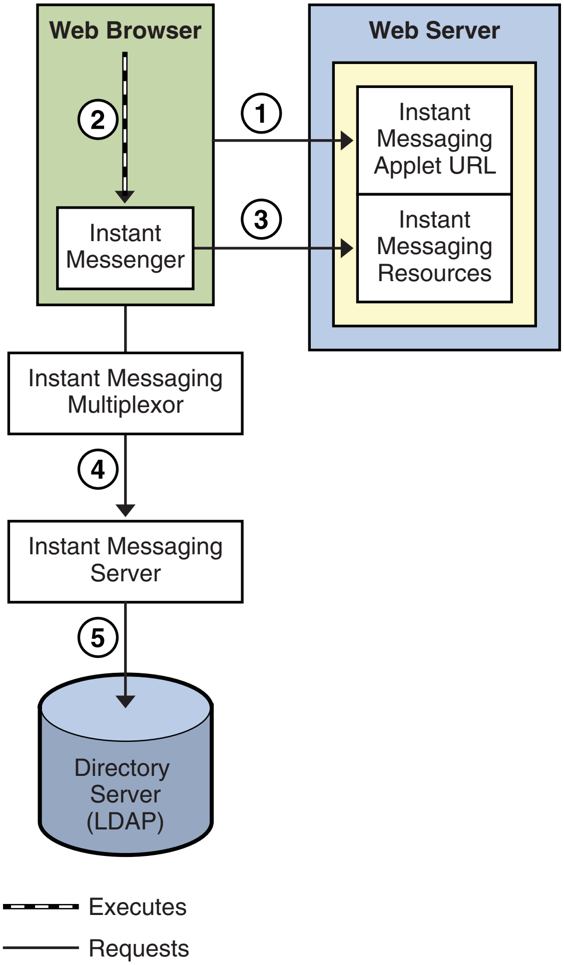

FIGURE 21–2 Flow of Authentication Requests in a Basic Instant Messaging Architecture

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 261. To provide this functionality, you need to include the components listed in “Basic Instant Messaging Architecture” on page 261. 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 21–3 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 21–4 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 263 for more information.

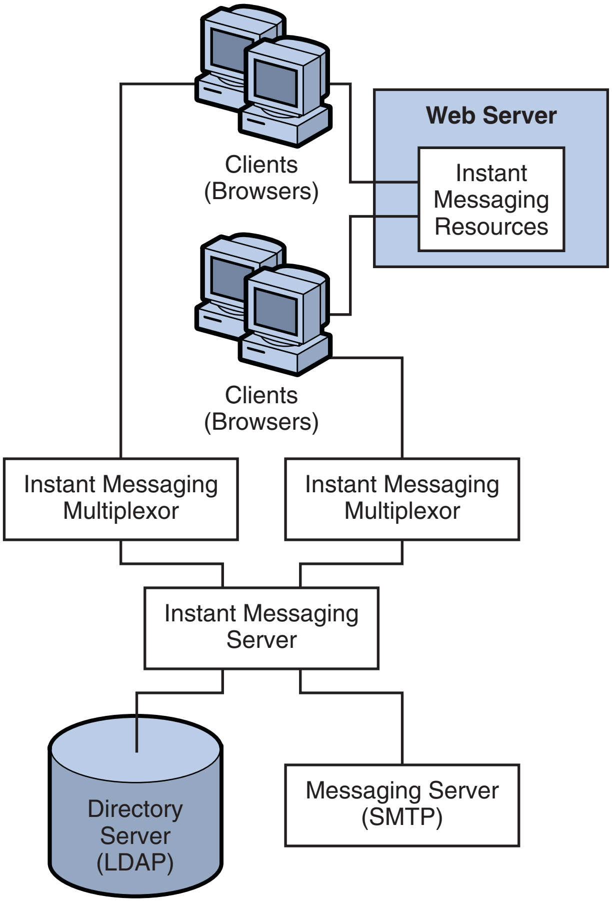

FIGURE 21–3 Instant Messaging Architecture with Email Notification

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.

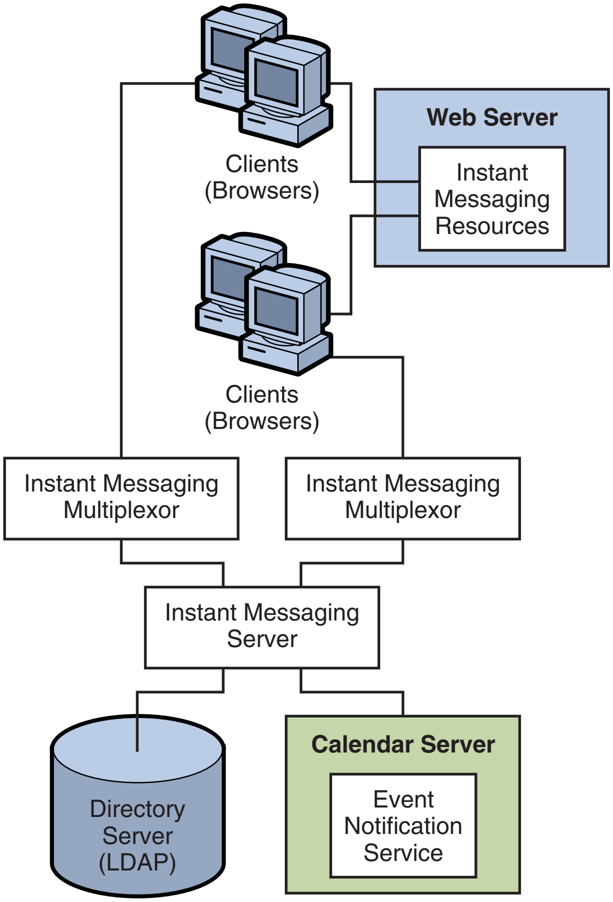

FIGURE 21–4 Instant Messaging Architecture with Calendar Alerts

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 261. To provide this functionality, you need to install the components listed in “Basic Instant Messaging Architecture” on page 261 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.

Figure 21–5 shows the Instant Messaging Access Manager architecture.

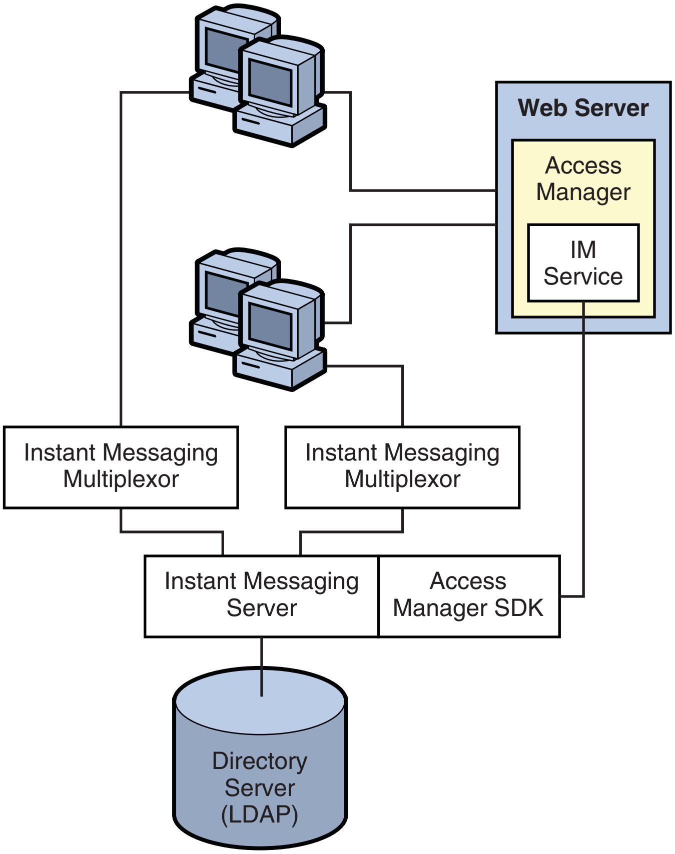

FIGURE 21–5 Instant Messaging Architecture With Access Manager-based Server Policy Management or Single Sign On

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 21–6 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 21–2, this figure focuses on the flow of authentication requests. An explanation of the steps in this process follows the figure.

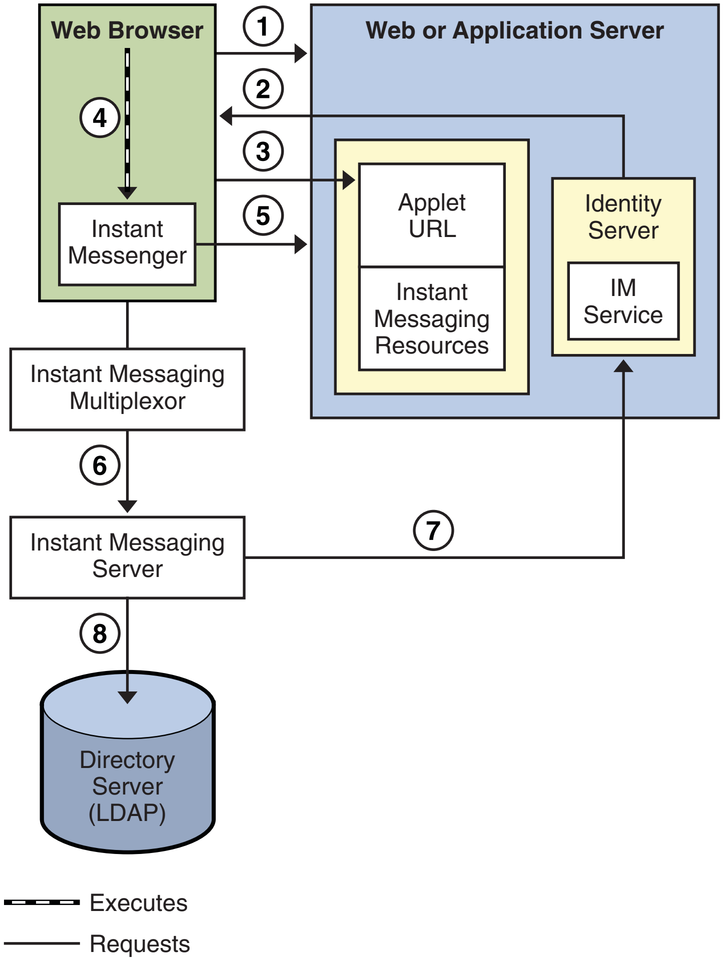

FIGURE 21–6 Flow of Authentication Requests in an Access Manager Configuration

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 have basic Instant Messaging functionality through the Portal Desktop. An Instant Messaging architecture that provides for this functionality also provides the same functionality as "Basic Instant Messaging Architecture" on page 261. 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 261 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 268 are also available.

Figure 21–7 shows the Instant Messaging Portal-based architecture.

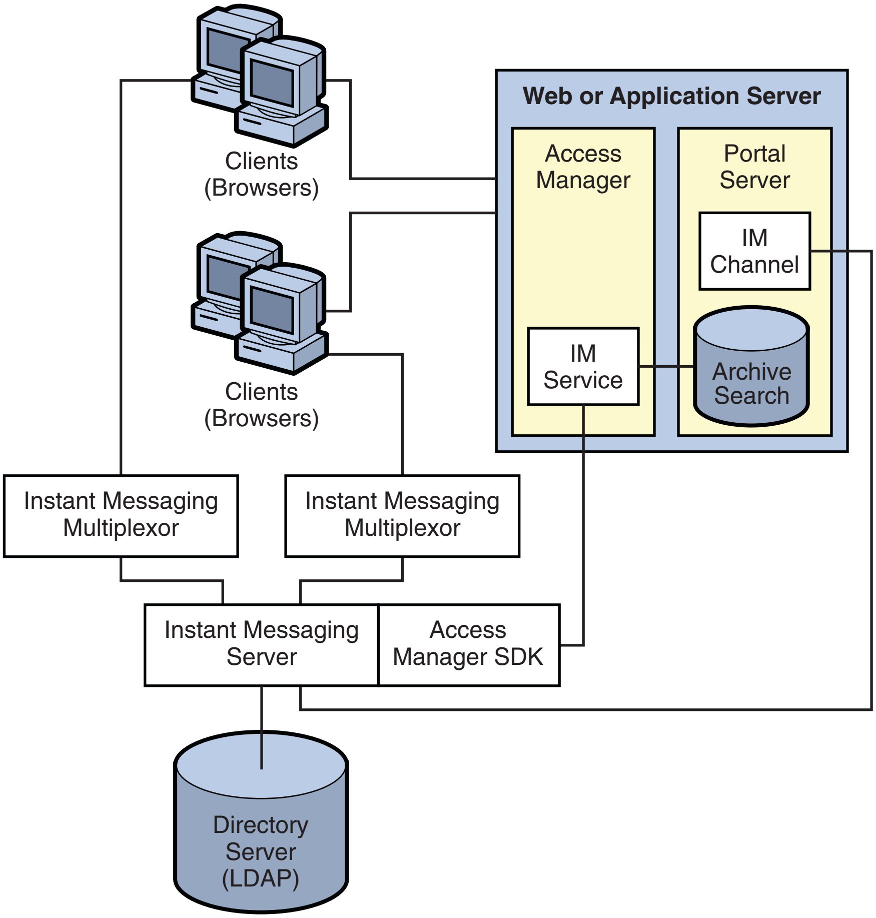

FIGURE 21–7 Instant Messaging Architecture With Portal-based Secure Mode or Archiving

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 21–8 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 21–2, this figure focuses on the flow of authentication requests. An explanation of the steps in this process follows the figure.

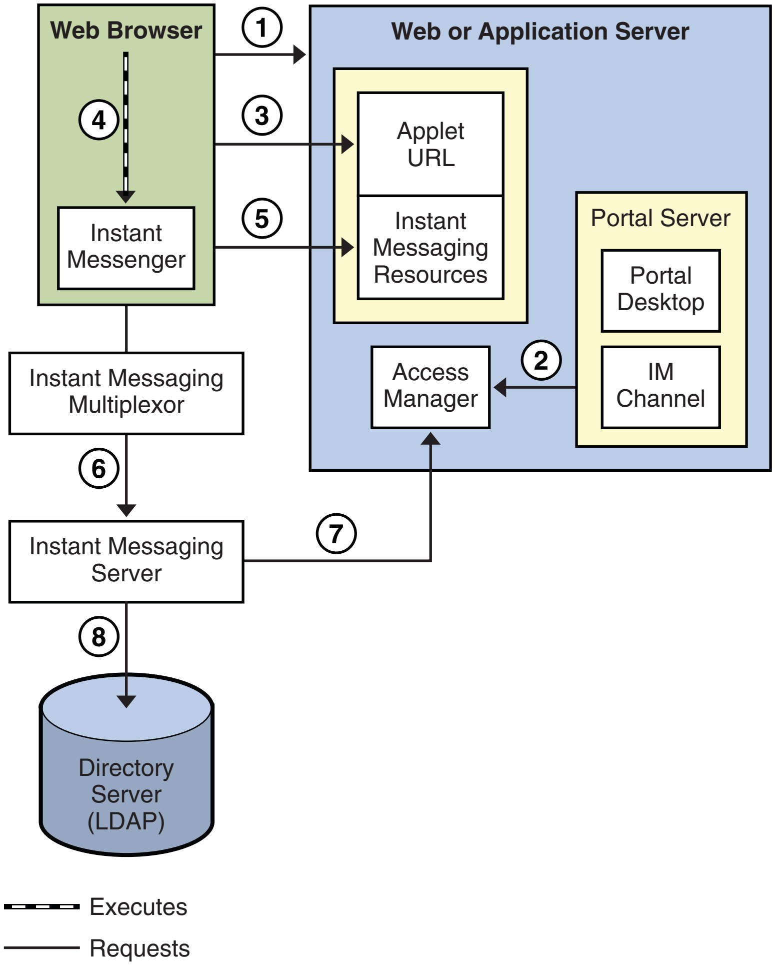

FIGURE 21–8 Flow of Authentication Requests in a Portal Server and Access Manager Configuration

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 261. 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" on page 276
- "Instant Messaging Physical Deployment Example: Multiplexors on Separate Hosts" on page 276
- "Instant Messaging Physical Deployment Example: Multiple Instant Messaging Hosts" on page 277

- 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 21–9 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.

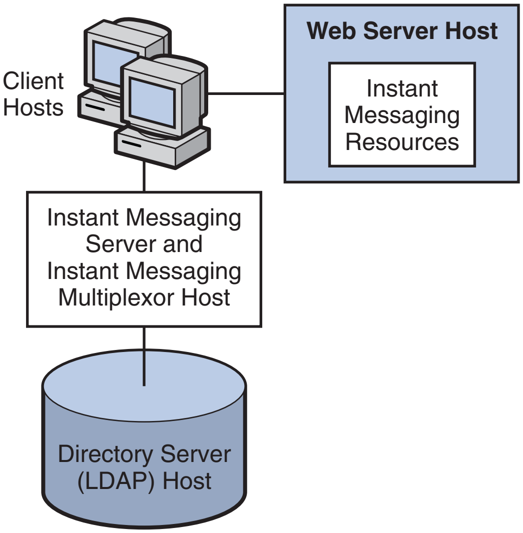
Client Hosts
Web Server Host
Instant Messaging Resources
Instant Messaging Server and Instant Messaging Multiplexor Host
Directory Server (LDAP) Host

FIGURE 21–9 Separate Web Server and Instant Messaging Hosts

Instant Messaging Physical Deployment Example: Multiplexors on Separate Hosts

Figure 21–10 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.

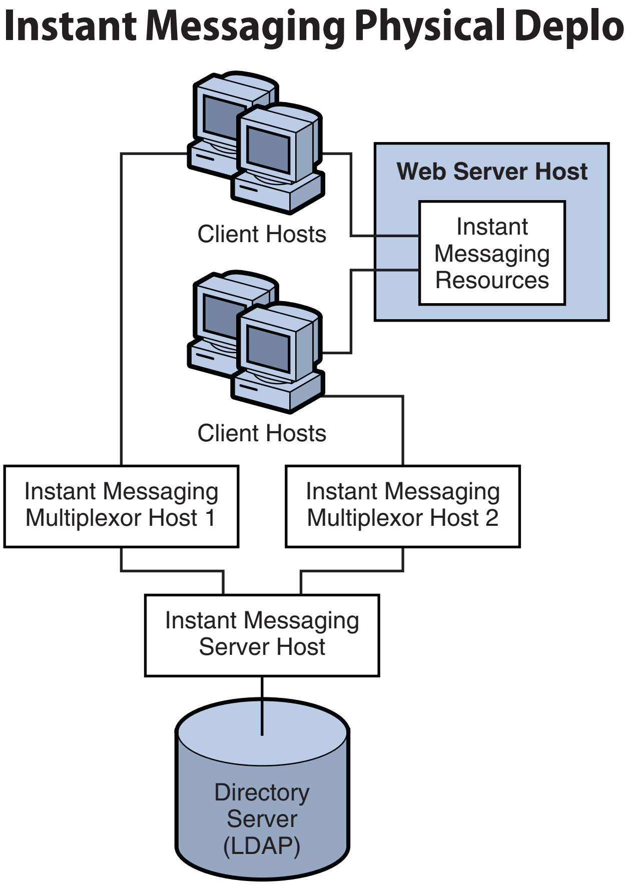
Instant Messaging Physical Deployment Example:
Client Hosts
Web Server Host
Instant Messaging Resources
Client Hosts
Instant Messaging Multiplexor Host 1
Instant Messaging Multiplexor Host 2
Instant Messaging Server Host
Directory Server (LDAP)

FIGURE 21–10 Instant Messaging Multiplexors Installed on Separate Hosts

Multiple Instant Messaging Hosts

Figure 21–11 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.

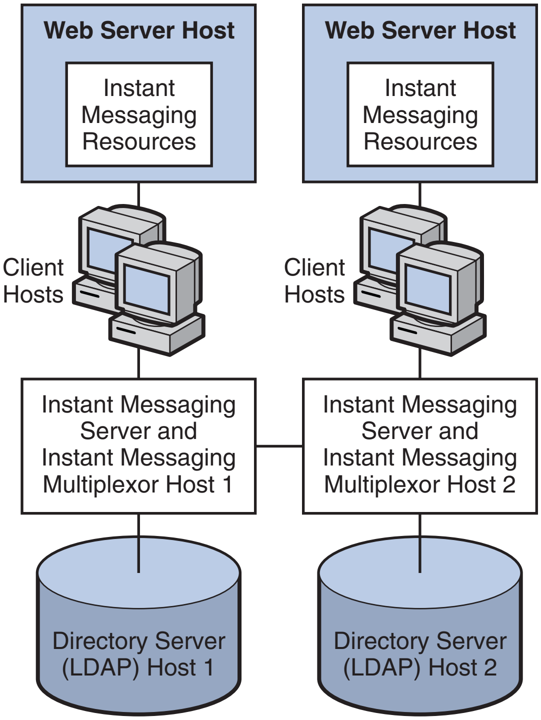
Web Server Host
Instant Messaging Resources
Web Server Host
Instant Messaging Resources
Client Hosts
Client Hosts
Instant Messaging Server and Instant Messaging Multiplexor Host 1
Instant Messaging Server and Instant Messaging Multiplexor Host 2
Directory Server (LDAP) Host 1
Directory Server (LDAP) Host 2

FIGURE 21–11 Multiple Instant Messaging Server Hosts

CHAPTER 22

Planning Instant Messaging Security

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

This chapter contains the following sections:

- "Protecting Instant Messaging Components in Your Deployment" on page 279
- "Planning Instant Messaging User Authentication" on page 282
- "Planning Instant Message Privacy, Security, and Site Policies" on page 283

Protecting Instant Messaging Components in Your Deployment

This section describes how to secure components in your Instant Messaging deployment.

Overview of Instant Messaging Security

Instant Messaging supports the following levels of security:

- **No Security.** Communication is all plain text from client through the multiplexor to the server.
- **Legacy SSL.** Secure communication between client and multiplexor, and plain text between multiplexor and server. (This is only relevant if you are using the multiplexor).
- **startTLS.** End-to-end secure communication between client and server. If you enable the multiplexor, it does not process any data but instead passes it on to the server.

The startTLS option enables end-to-end encryption (the communication between client-multiplexor-server is all in encryption form), while legacy SSL enables encryption between the Instant Messenger client up to the multiplexor: the communication between multiplexor and server is in plain text (though in a proprietary protocol). Use startTLS if you require a higher level of security. If you use startTLS, you do not need an alternate means of securing the multiplexor-to-server communication (it will be secure).

Protecting Instant Messaging Server and Multiplexor

Instant Messaging supports TLS (Transport Layer Security) and legacy SSL (Secure Sockets Layer) for secure communications. Instant Messaging uses a startTLS extension to the Transport Layer Security (TLS) 1.0 protocol for client-to-server and server-to-server encrypted communications and for certificate-based authentication between servers. In addition, Instant Messaging supports a legacy implementation of the SSL protocol (version 3.0) for encrypted communications between the Instant Messenger client and the multiplexor.

When planning for SSL for Instant Messaging, keep in mind the following:

- You can secure the Instant Messaging deployment by enabling SSL on the web container port (either Web Server or Application Server) and accessing Instant Messaging functionality by using the XMPP/HTTP Gateway (httpbind). You can also enable SSL at the mutliplexor and the Instant Messaging client can access Instant Messaging functionality directly through the mutliplexor.
- The Instant Messenger client uses only legacy SSL if it is enabled at the multiplexor. In this case, the Instant Messenger client does use startTLS of the server.
- Instant Messaging does not support end-to-end encryption using legacy SSL, so the communications between the Instant Messaging server and multiplexor is in the clear if the Instant Messaging server and multiplexor are deployed on multiple nodes and SSL is enabled at the multiplexor. If you want end-to-end encryption, you must use startTLS.
- Set the proper file and directory permissions for Instant Messaging configuration files (`iim.conf` and `httpbind.conf` in the `/etc/opt/SUNWiim/default/config` directory).[1] When you create additional Instant Messaging server/multiplexor instances manually, you need to also ensure that the file and directory permissions are correct. The default installation sets the file/directory permissions. The default instance directory (`/var/opt/SUNWiim/default`) has the following permissions:

  ```
  drwx------   5 root     other        512 Oct 16 14:24 default
  ```
- Take care when enable Instant Messaging monitoring as this exposes server statistics that could be considered security issues. The default configuration does not enable the monitoring feature. You enable this property through the `iim.conf` file.
- Enable debug logging only when needed, as this impacts overall system performance. Though passwords are not logged, the protocol communication between users is logged, which could be a potential security issue.

[1] Instant Messaging runs as the user specified in the `iim.conf` file. This user needs read access to the file. If you use httpbind, the user that runs the web container should be able to access the Instant Messaging directory path and configuration file. Typically , if the deployment of Instant Messaging includes Access Manager or Portal Server, the default user is `root`.

- When you enable startTLS, use a single server certificate for both client-to-server and server-to-server communication.
- The Instant Messaging server should be able to connect to Access Manager if identity integration is enabled. Similarly, Instant Messaging server needs access to the Portal Server search when deployed with Portal Server. Additionally, an Instant Messaging deployment that leverages LDAP needs proper authentication for access. Refer to each product's individual documentation to understand how to secure each component.

The Instant Messaging default installation supports only SASL Plain. If you require a higher level of security, use the Instant Messaging public Service Provider Interface. SASL Plain and jabber:iq:auth are two forms of plain text authentication. That is, in both, the password is sent in the clear (encoded perhaps, but still clear text) and so both are insecure forms of authentication. Nevertheless, this is an issue only if end-to-end encryption (through startTLS for direct socket connection, and HTTPS for httpbind) is not enabled. If end-to-end encryption is enabled, the password is not “seen” in the clear on the network.

Alternatively, if you do not want to transmit passwords in the clear (even if over an encrypted stream), use the Instant Messaging SPI for plugging in authentication mechanism's at the server side through SASLRealm. You can implement custom SASL mechanisms as implementations but you will then need an Instant Messaging client that supports this custom mechanism. The Sun Java System Instant Messaging client supports only SASL Plain, jabber:iq:auth (both insecure).

For more information, see Chapter 12, “Securing Instant Messaging Using TLS and Legacy SSL,” in *Sun Java System Instant Messaging 7.2 Administration Guide*.

Providing Instant Messaging Client Access Through a Firewall

The XMPP/HTTP Gateway (httpbind) provides Instant Messaging access to XMPP-based clients, such as HTML based clients, and clients that are behind firewalls that allow HTTP traffic only and don't permit XMPP traffic. The gateway proxies Instant Messaging traffic to the XMPP server on behalf of HTTP clients.

When planning to use the XMPP/HTTP Gateway, keep in mind the following:

- Use port 5222 at the Gateway if the Gateway is communicating to the server through a multiplexor. Also, use port server port 5269 if no multiplexor is involved. You can specify port 5222 or 5269 in the `httpbind.conf` file.
- The XMPP/HTTP gateway supports both startTLS and legacy SSL. If you want legacy SSL support, enable SSL on the Web Server port. However, if the XMPP/HTTP gateway configuration points to the multiplexor instead of the server, disable legacy SSL at the multiplexor. Additionally, if you want startTLS support, enable startTLS on the server and all communications will be encrypted.

- Do not expose the Instant Messaging server to direct access. In a typical deployment scenario you would locate the multiplexor in the DMZ, and open the multiplexor to server communication port (45222 usually) in the second firewall.

Protecting the Instant Messaging Archive

Instant Messaging has the capability to archive instant messages for later retrieval and searching. If you enable the email archive, you need to decide which administrators will receive email containing archived instant messages. You can configure a separate list of administrators to receive polls, news, conference, alerts, or chat sessions. You can also configure Instant Messaging to use the extended RFC 822 header. Doing so allows mail clients to filter messages based on the header content. If you do enable the Portal archive, you can decide which administrators can access the Portal archive database.

For more information, see Chapter 18, "Managing Archiving for Instant Messaging," in *Sun Java System Instant Messaging 7.2 Administration Guide*.

Planning Instant Messaging User Authentication

User authentication enables your users to log in through their Instant Messaging clients to chat and access other features of Instant Messaging.

Instant Messaging and Passwords

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 Instant Messaging administration. See the Directory Server documentation to understand directory server password policies:

http://docs.sun.com/coll/1316.1

Instant Messaging and LDAP

All deployments of Instant Messaging require a directory server. In a deployment without Access Manager, the Instant Messaging server uses the directory server to perform end-user authentication and to search for end users. For various ways to secure the directory, see the Directory Server documentation.

In a deployment with Portal Server, the Instant Messaging server uses the directory used by Portal Server. When installed in an Access Manager deployment environment, the Instant Messaging server uses the directory used by the Access Manager to search for end users, and not for end-user authentication. In an Access Manager deployment, Access Manager performs the authentication.

If you use an LDAP directory to maintain your user namespace, the default configuration makes the following assumptions regarding the schema used by this directory:

- End user entries are identified by the `inetOrgPerson` object class.
- Group entries are identified by the `groupOfUniqueNames` or `groupofURLs` object class.
- Instant Messenger user ID attribute of an end user is provided by the `uid` attribute (from `inetOrgPerson objectclass`).
- The email address of an end user is provided by the `mail` attribute.
- The display name of an end user or group is provided by the `cn` attribute.
- The list of members of a group is provided by the `uniqueMember` attribute (`groupOfUniqueNames` object class).

Note – Some user attributes might contain confidential information. Ensure that your directory access control is set up to prevent unauthorized access by non-privileged users.

Instant Messaging and Searching the Directory Anonymously

Instant Messaging needs to be able to search the directory to function correctly. You need to ensure that your directory is configured to be searchable by anonymous users. If your directory is not readable or searchable by anonymous users, you must take configuration additional steps.

For more information, see Chapter 11, “Managing Instant Messaging’s LDAP Access Configuration,” in *Sun Java System Instant Messaging 7.2 Administration Guide*.

Planning Instant Message Privacy, Security, and Site Policies

Instant Messaging provides the ability to control access to Instant Messaging features and preserve end-user privacy.

Instant Messaging Site Policies

Site policies specify end-user access to specific functionality in Instant Messaging. When developing your site policies for Instant Messaging, keep in mind the following questions:

- Can users access the presence status of other end users?
- Can users send alerts to other end users?
- Do you want users to save properties on the server?
- Who do you want to be able to create and manage conference rooms?

- Who do you want to be able to create and manage news channels?

For more information, see Chapter 17, “Managing Instant Messaging and Presence Policies,” in *Sun Java System Instant Messaging 7.2 Administration Guide*.

Methods of Controlling Instant Messaging End User and Administrator Privileges

Different sites using Instant Messaging server have different needs in terms of enabling and restricting the type of access end users have to the Instant Messaging service. The process of controlling end user and administrator Instant Messaging server features and privileges is referred to as policy management. You choose from two methods of policy management: through access control files or through Sun Java System Access Manager.

- **Managing Policies Using Access Control Files.** The access control file method for managing policies allows you to adjust end-user privileges in the following areas: news channel management, conference room management, the ability to change preferences in the User Settings dialog, and ability to send alerts. It also allows specific end users to be assigned as system administrators.
- **Managing Policies using Sun Java System Access Manager.** This method gives you control of the same privileges available with the access control file method; however, it additionally allows more fine-tuned control over various features, such as the ability to receive alerts, send polls, receive polls, and so forth. Furthermore, managing policies using Sun Java System Access Manager gives you finer-tuned control over privileges.

If your deployment does not include Access Manager, you must use the access control file method to manage policies. If you are using Access Manager with Instant Messaging, and you have installed the Instant Messaging and Presence services components, you can use either of the policy management methods. Managing policies using Access Manager is a more comprehensive method. One advantage of this method is that it allows you to store all end-user information in the directory.

For more information, see Chapter 17, “Managing Instant Messaging and Presence Policies,” in *Sun Java System Instant Messaging 7.2 Administration Guide*.

PART V

Deploying Communications Express

This part contains the following chapters:

- Chapter 23, “Introduction to Communications Express Software,”
- Chapter 24, “Developing a Communications Express Architecture,”

CHAPTER 23

Introduction to Communications Express Software

Communications Express provides an integrated web-based communications and collaboration client. Communications Express is a common component 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" on page 287
- "Communications Express Features" on page 288
- "Communications Express High-Level Architecture" on page 288

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). The Messaging Server packages can then be configured to run as Webmail server, which connects to a back-end store running Webmail server. Also, both Communications Express and Messenger Express can run on different IP addresses.

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 System Communications Express 6.3 Administration Guide* for more information.

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, the `mshttpd` daemon can be local or remote, and the LDAP service can all 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 automatically established because both applications are deployed within a single web container. Also, you can optionally use Access Manager to provide the 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 System Communications Express 6.3 Administration Guide* for more information.
- 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 any web container. The mail module is rendered by the Messenger Express. Messenger Express is the standalone web interface mail application that uses the HTTP service of the Messaging Server.

Figure 23–1 shows the Communications Express software architecture.

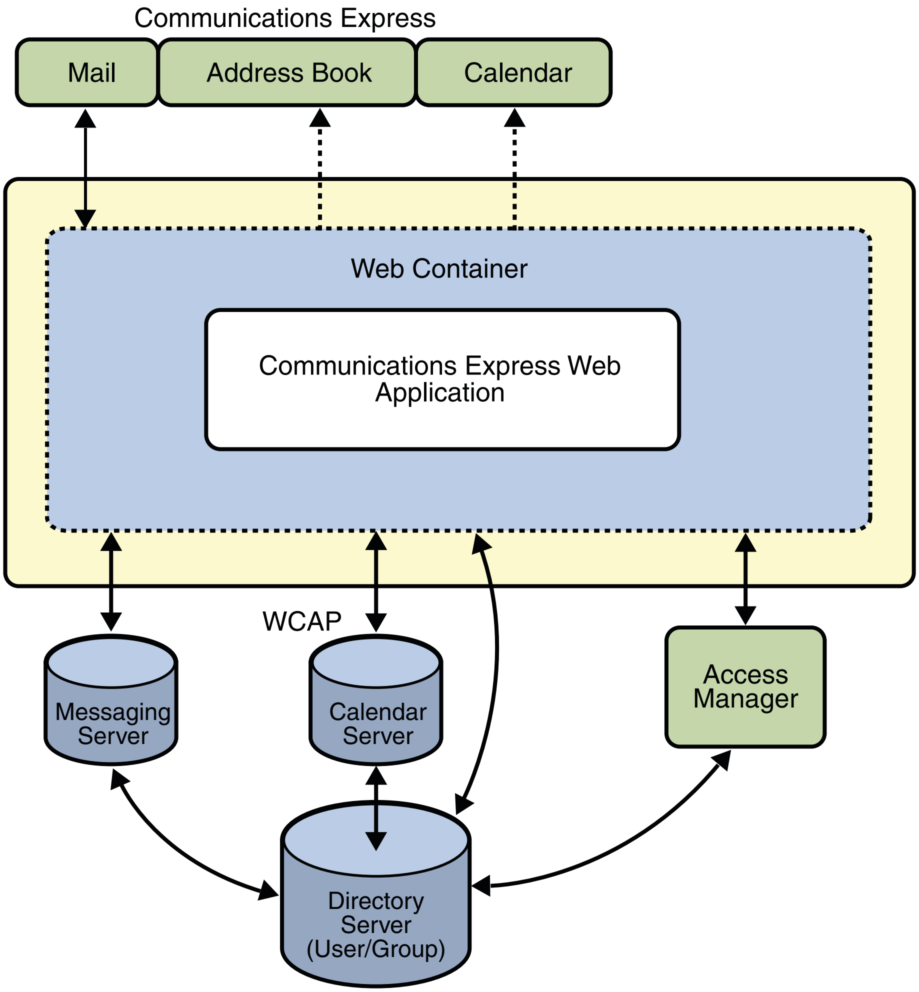

FIGURE 23–1 High Level Communications Express Software Architecture

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 24

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" on page 291
- "Communications Express on Remote Host Architecture" on page 293

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.

Figure 24–1 shows the basic Communications Express architecture.

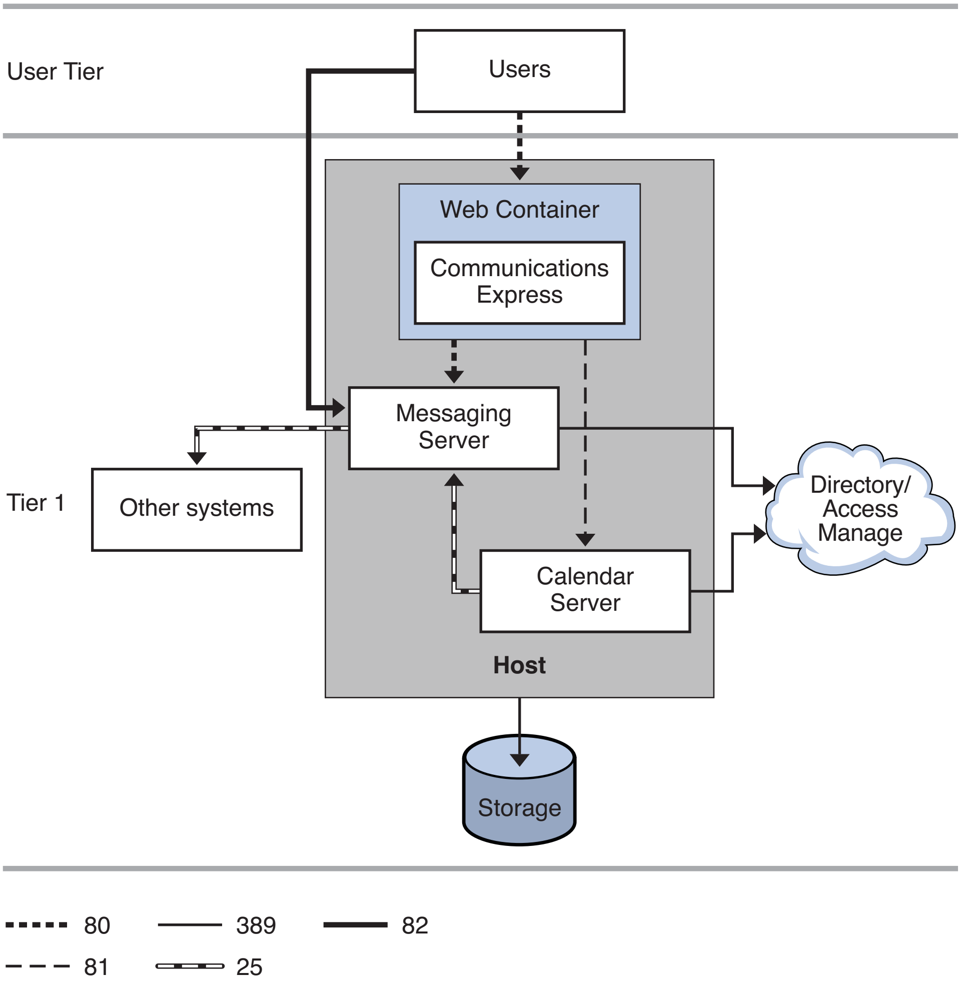

FIGURE 24–1 Basic Communications Express Architecture

The following table explains the protocols and port numbers used by this architecture.

TABLE 24–1 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

TABLE 24–1 Protocols And Ports Used by Basic Communications Express Deployment Architecture *(Continued)*

Protocol	Port	Used By
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 24–2 shows a Communications Express architecture where both intranet and Internet users log on to Communications Express on the front-end host, which accesses the back-end host on the user's behalf. You can enable SSO on the front-end host such that Communications Express validates the session in Access Manager.

You install the front-end host with the following components:

- Messaging Server (Webmail Server)
- Calendar Server
- Web container
- (Optional) Access Manager SDK
- Communications Express

You install the back-end with the following components:

- Directory Server
- Messaging Server (Mail Store)
- Calendar Server
- (Optional) Web Container and Access Manager

Figure 24–2 shows the Communications Express on remote host architecture.

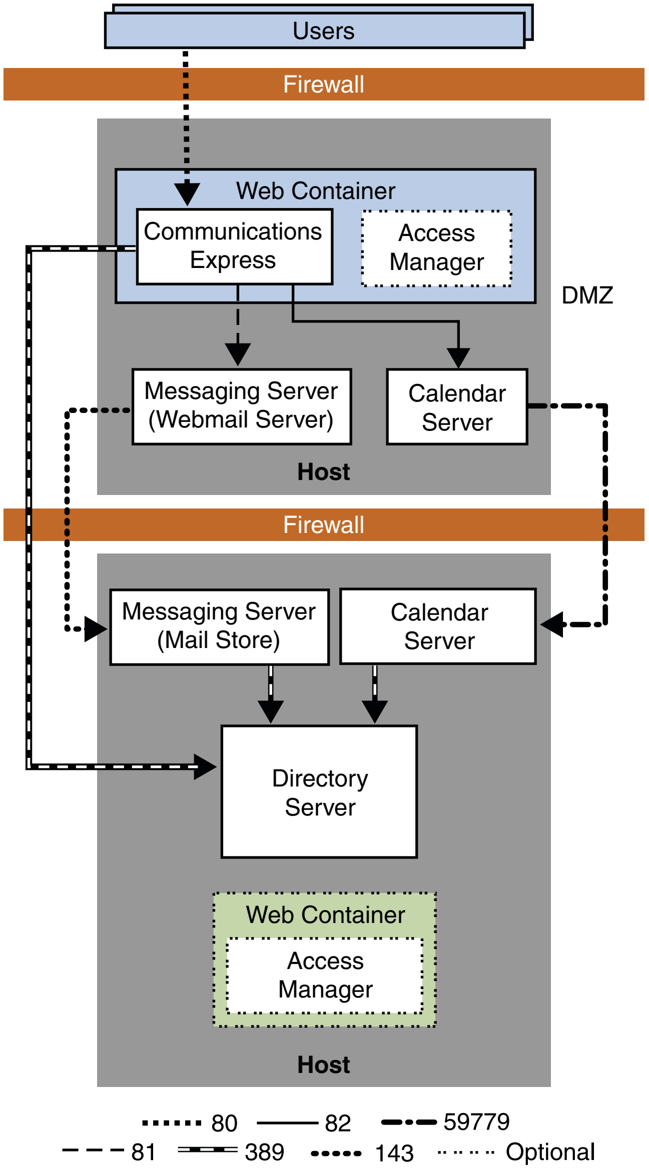

FIGURE 24–2 Communications Express on Remote Host Architecture

The following table explains the protocols and port numbers used by this architecture.

TABLE 24–2 Protocols And Ports Used by Communications Express Remote Host Deployment Example

Protocol	Port	Used By
HTTP	80	Users to communicate with Communications Express on the front-end host in the DMZ
HTTP	81	Communications Express on the front-end host in the DMZ to communicate with Webmail Server also on the front-end host
HTTP	82	Communications Express on the front-end host in the DMZ to communicate with Calendar Server also on the front-end host
IMAP	143	Webmail Server on the front-end host in the DMZ to communicate with Messaging Server on the back-end host
LDAP	389	Messaging Server, Calendar Server, and Communications Express to communicate with LDAP directory
WCAP	59779	Calendar Server on the front-end host in the DMZ to communicate with Calendar Server on the back-end host

PART VI

Deploying Connector for Microsoft Outlook

This part contains the following chapters:

- Chapter 25, “Introduction to Connector for Microsoft Outlook Software,”
- Chapter 26, “Developing a Connector for Microsoft Outlook Architecture,”
- Chapter 27, “Understanding Connector for Microsoft Outlook Pre-Installation Considerations,”

CHAPTER 25

Introduction to Connector for Microsoft Outlook Software

Connector for Microsoft Outlook enables Outlook to be used as a desktop client on Windows with Messaging Server and Calendar Server, with support for Address Book as well.

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 WCAP 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 includes support for address book service. You must install Communications Express to enable the address book service. This service makes use of WABP and allows a user to have their personal address book accessible from both Outlook and Communications Express.

This chapter contains the following sections:

- "Connector for Microsoft Outlook Overview" on page 300
- "Connector for Microsoft Outlook Features" on page 300
- "Connector for Microsoft Outlook High-Level Architecture" on page 302

Connector for Microsoft Outlook Overview

Organizations deploy Connector for Microsoft Outlook so that users can use Microsoft Outlook as their email and calendar client while connected to Sun Java Enterprise System servers. The Connector for Microsoft Outlook plug-in must be installed and configured for each desktop user to facilitate the necessary ongoing communications between Outlook and the servers. Also provided is the desktop deployment toolkit. The toolkit is a collection of software tools for system administrators designed to simplify both the administrator's work and the user tasks associated with installing and configuring the Connector for Microsoft Outlook plug-in for individual users.

The desktop deployment toolkit allows the administrator to create customized end-user installation packages with preset configuration parameters to simplify and streamline the user's process. This also enforces any configuration settings the administrator deems necessary or desirable for a particular user or group of users. The deployment configuration program saves those preset configuration parameters in an `.ini` text file, and then bundles the `.ini` file with an installation program—the setup wizard—for end users. When an end user activates the package, the setup wizard reads the `.ini` file to install and configure the Connector software on his or her desktop according to the administrator's specifications.

Connector for Microsoft Outlook Features

The following key features and functionality are available with Connector for Microsoft Outlook:

- Access to Messaging Server, Calendar Server, and Address Book Server
- Access to email folders on the Messaging Server via IMAP4
- Access to calendar data (events, appointments) and tasks stored in the Calendar Server via WCAP
- Access to corporate directory via LDAP
- Access to contacts stored in the Address Book server by using WABP
- Access to multiple calendars and address books
- Typical Outlook mail features:
  - Compose, reply, and forward messages
  - Use Microsoft Word to write and edit messages
  - Apply spell check and encryption to message body
  - Apply address completion to message headers
  - Add signatures to messages
  - Polling multiple folders
- Share mail folders with other users and set specific permissions for shared contacts
- Subscribe to other users' mail folders

- Typical calendar features:
  - Creating and sharing multiple calendars
  - Creating and sharing multiple address books
  - Viewing group memberships within contact details
  - Access attachments in calendar events and todos
  - Create new appointments and events (all-day, recurring, public, private)
  - Modify events
  - Check availability
  - Suggest alternate time for events
  - Track responses to event requests
  - Create and share multiple address books
  - View group memberships within Contact details
- Email based group scheduling
- Share calendar with other users and set specific permissions for shared calendars
- Subscribe to other users' calendars
- Delegate calendar access to others
- Typical task-related functionality (new task, modify task)
- Client-side rules attached to folders and data types
- Notes and Journal stored in IMAP folders
- Full offline support for mail, calendar, and address book
- Interoperability with web client (Communications Express)
- Management of server-side message filters
- Global Address List with browsing capabilities
- Removal of dependency on Microsoft LDAP
- Out-of-office setting, and ability to notify user that setting is activated at login
- Progress displayed when synchronizing large folders
- Setting mail filters
- Sharing contacts with other users and setting specific permissions for shared contacts
- Ability to search for users for mail/Calendar/Contacts folders sharing
- Name and email address automatically set and updated from LDAP
- Option to preview body of all messages

Connector for Microsoft Outlook High-Level Architecture

The Sun Java System Connector desktop deployment toolkit consists of the following components:

- **Deployment Configuration Program** allows system administrators to create bundled, customized installation packages for end users. These packages can then install and configure Connector for Microsoft Outlook and, if appropriate, convert data from Microsoft Exchange local stores.
- **Setup Wizard** allows end users to install the Connector for Microsoft Outlook plug-in, configure its operations and features based on the administrator's settings, and convert existing Outlook data files (`.pst` files) associated with Exchange into a form that Connector for Microsoft Outlook can use. (The setup wizard is one part of the bundled end-user installation package created by the deployment configuration program.)
- **Install kit (MSI)** is the installation utility for the software that facilitates ongoing, permanent communications between Microsoft Outlook features and the Sun Java Enterprise System servers. This is supplied as part of the installation package.
- **Special Kits for using Microsoft System Management Services (SMS) to install the Connector for Microsoft Outlook** support the “push” functionality of Microsoft's SMS to let a system administrator distribute and install the desktop components to user desktops with minimal or no user involvement.

In addition to the above-listed components of the deployment toolkit, the **Microsoft installation program for Web Publishing Wizard (WPW)** (available from other vendors) may be necessary for your users to install the Sun Java System Connector for Microsoft Outlook.

Microsoft's WPW is a tool used to generate HTML-coded documents, typically in the form of web pages to be viewed in a web browser. But WPW can also be used to “publish” information for other purposes—to create documents containing certain types of data and upload the documents to locations where other applications can find them and extract the pertinent data. Connector for Microsoft Outlook uses WPW this way to facilitate the transfer of users' free/busy data between Outlook and the Sun Java Enterprise System servers, and WPW therefore is a necessary intermediary and a required component of every user's installation.

Note – The WPW, however, is not included in the deployment toolkit package because it is a Microsoft product, so you must obtain the Web Publishing Wizard from Microsoft. The installation program for Microsoft’s WPW is available as a free download from `http://www.microsoft.com`.

◆ ◆ ◆ CHAPTER 26

Developing a Connector for Microsoft Outlook Architecture

This chapter introduces the deployment process of the Connector for Microsoft Outlook.

This chapter contains the following sections:

- "Desktop Deployment Process Tasks" on page 303
- "Why Use the Deployment Toolkit?" on page 304

Desktop Deployment Process Tasks

Deployment of the Connector for Microsoft Outlook plug-in to each user desktop requires three distinct tasks:

- **Installation.** The necessary and appropriate software must be physically installed to the Outlook user's desktop. Software installation requires access privileges that often are disallowed to many or most end users. In this case, most enterprises implement a "push" method for software distribution from the system administrator to user desktops that bypasses the requirement for user access privileges. (This "push" method of distribution is explained in more detail in "Desktop Installation Methods.") If your network serves "locked-down" Windows environments where end users cannot install software, we strongly recommend this sort of automated configuration management as a way to avoid many individual desktop visits.
- **Configuration.** Connector for Microsoft Outlook is installed with an assortment of configuration parameters including the server names and port numbers, user password options, directory search defaults, the log file path, and so forth. While users or administrators can manually configure these settings within Outlook at each desktop, it is far more efficient for administrators to pre-configure these for groups of users and avoid desktop visits.
- **Conversion.** The desktop setup wizard can convert Exchange users' Contacts, Journal, and Notes data to local (desktop) Personal Folders (`.pst`) files. Any such existing personal data files associated with Microsoft Exchange and Outlook must be converted to be compatible with the plug-in. (This task does not apply to new email users who were not previously

working with Microsoft Exchange.) If a user has password-protected his or her data files, these conversions will require the password associated with the protected files. The conversion function is processor-intensive and may run on the user's computer for many minutes or even hours if a user has large amounts of data to convert. For this reason the conversion utility lets a user defer the conversion of larger files to a later time, such as during a lunch break or even overnight.

The Deployment Configuration Program lets a system administrator create installation packages that will automate some or all of these tasks for end users, depending on the administrator's deployment strategy for any particular group of users.

Why Use the Deployment Toolkit?

With the deployment toolkit, an administrator can control a wide range of configuration parameters for desktop users. Mandating many or most configuration settings will bypass the need for users to ponder options, make choices and set values themselves. These automated or semi-automated installations will spare the corporate help desk many calls for guidance, support, and solutions to the inevitable problems that arise when user choices produce unexpected results. Overall, the toolkit substantially reduces the cost, time and effort required to deploy the Sun Java System Connector software.

A system administrator may create different installation packages for different groups of desktop end user—for example, to enforce different configuration schemes for users in the Sales department versus the Engineering department and so forth, or to offer configuration options to some groups of users while setting fixed parameters (eliminating the choices) for other groups.

If the organization is migrating from Microsoft Exchange, the user's installation packages will also preserve the considerable value of their existing Outlook data stored in `.pst` files on user desktops, as well as their Notes, Journals and Contacts stored on the Exchange server. The bundled installation packages include a conversion utility that quickly converts all such data to pure Internet addresses, so users will be able to reply to older messages, appointment invitees will receive change notices, and address books and personal distribution lists will remain serviceable after your migration to the Java Enterprise System server.

CHAPTER 27

Understanding Connector for Microsoft Outlook Pre-Installation Considerations

This chapter describes considerations you need to think about before installing Connector for Microsoft Outlook.

This chapter contains the following sections:

- "Connector for Microsoft Outlook Installation Considerations" on page 305
- "Exchange to Communications Suite Migration Considerations" on page 308
- "Where to Go for More Connector for Microsoft Outlook Information" on page 310

Connector for Microsoft Outlook Installation Considerations

The deployment toolkit contains a variety of deployment options with operational options that permit considerable flexibility in devising and implementing a suitable migration strategy for most any environment, circumstances and administrator preferences. The topics in this section describe the most common scenarios, and explain how the migration tools accommodate them.

Interactive User Installation (Self-Service)

The setup wizard is designed to be straightforward for end users to run by themselves. The setup wizard can be placed on a file server, so it need not be individually installed on end-user workstations. However, the physical installation of the plug-in on user desktops requires access privileges that often are disallowed to many or most end users. If users do not have installation privileges for their own desktops, either of the following strategies may be used:

- Administrators physically install the plug-in on individual users' desktops.
- Use a configuration management tool to "push" the software to multiple users' desktops (explained in "Automated Installation Using Configuration Management Tools" on page 306).

After the software has been physically copied to a user's desktop, the user can run the setup wizard to configure the software and convert existing Personal Folders (`.pst`) files.

End users who run the conversion program will provide their own credentials for the servers. This method therefore permits conversion of password-protected `.pst` files (see "Password-Protected Personal Stores in Outlook" on page 307 below), and lets users specify which of their personal stores should be converted for use with the plug-in.

Users can read unconverted email messages, but cannot reply to them because unconverted addresses are unfamiliar to the new server. Users who have some personal stores that are very old, so that the need for a future reply is highly unlikely, may therefore opt to leave such files unconverted. The conversions can run in the background, freeing the user's computer for other work, but the process is likely to slow the performance of other applications.

The significant downsides of interactive user installation are:

- Increased demand for support from your organization's help desk, which may be considerable depending on users' technical skills and the complexity of "before" and "after" network configurations.
- Time and effort the administrator has to devote to visiting multiple user workstations to physically copy the software to the users' desktops (for users who are not authorized to perform that task for themselves).

Individual Desktop Installation

Administrators may allow some users to perform their own installations, as described above, but visit other user's desktops to perform some or all of the installation and configuration tasks for them. This approach can ensure a smooth migration for top executives or less technical users who are not prepared to perform the tasks for themselves. The deployment plan should address whether these sorts of administrator visits are warranted for any users in the organization, and for whom.

Automated Installation Using Configuration Management Tools

Software installation on user desktops requires access privileges that often are disallowed to many or most end users. Most administrators of such networks use a configuration management tool, such as Microsoft's SMS, to "push" the software to multiple users' desktops—a method that bypasses the requirement for user access privileges. If your network serves "locked-down" Windows environments, where end users cannot install software, this sort of automated configuration management can spare the administrator many visits to individual user desktops.

To accomplish a “push” distribution, you can use the deployment configuration program to build two different bundled installation packages for each user, to be executed in succession. The first would perform the “push” installation of the necessary software, while the second would run an interactive process by which the user could make choices about the configuration of the installed software and the conversion of the user's own existing data files. This “push” method may even be used to completely automate the conversion process for end users, but would require some scripting since the package must be invoked with information specific to each end user (e.g., the user's credentials).

The *Sun Java System Connector for Microsoft Outlook 7.2 Administration Guide* provides instructions for using Microsoft's SMS to implement this “push” method of software distribution. The *Administration Guide* also explains how to use command-line switches with an SMS script to fully automate the process by passing the necessary user passwords, for the user’s Personal Folders (`.pst`) files, to the desktop installation program.

Command-Line Switches for Desktop Installation

The Sun Java System Connector Setup Wizard supports command-line switches that may be used in combination with the other desktop installation methods described above, or with an SMS script as described in “Command-Line Switches for the User Installation Package” in *Sun Java System Connector for Microsoft Outlook 7.2 Administration Guide*.

The installation package will support these command-line switches:

`/USERNAME=`*xxx*, where *xxx* is the username on the Sun servers
`/PASSWORD=`*xxx*, where *xxx* is the password on the Sun servers
`/FULLNAME=`*xxx*, where *xxx* is the display name of the user
`/EMAILADDRESS=`*xxx*, where *xxx* is the email address of the user
`/DN=`*xxx*, where *xxx* is the user DN on the Sun servers
`/NEWPROFILENAME=`*xxx*, where *xxx* is the name of the created profile
`/SAVEPASSWORD=`*n*, where *n* = 1 (save) or 0 (don't save)

These switches will be useful if you are converting an Exchange profile:

`/OLDDOMAIN=`*xxx*, where *xxx* is the Exchange domain
`/OLDUSERNAME=`*xxx*, where *xxx* is the Exchange user name
`/OLDPASSWORD=`*xxx*, where *xxx* is the Exchange password

Password-Protected Personal Stores in Outlook

Outlook users can assign passwords to their Personal Folders (`.pst`) files, but the Sun Java System Connector Setup Wizard needs to open and modify these files in order to convert them for use with the new Connector software and the Sun Java System server. Your end users will therefore have to provide the passwords for any `.pst` files that they want to convert.

The Setup Wizard will automatically prompt users for the necessary passwords as they are needed, but obviously this will require user involvement that makes a Silent Mode setup impossible. If it is important for you to run the Setup Wizard in Silent Mode, users can be instructed to remove all such passwords during the conversion, or simply let the Wizard run with the passwords in place. If the Setup Wizard runs in Silent Mode and encounters a password-protected file, it will not convert the file, and will report that not all files were converted. Depending on the settings in your administrator's Deployment Configuration tool, the Setup Wizard may also log the event as an error.

Exchange to Communications Suite Migration Considerations

This section describes considerations to be made before migrating from a Microsoft Exchange server to Communications Suite.

Migration of Important Server Data: More Than Just Messages

The Exchange server contains valuable user information, and an efficient migration will extract and use this information to provision the user accounts on the new Communications Suite server. In addition to users' old mail messages, the old server contains user calendars, tasks, and personal address books and contacts. The old server also contains user names, primary Internet addresses, Internet aliases, phone numbers, postal addresses, and even descriptive information such as users' departments, titles, and so forth, and also all of the enterprise's public distribution lists.

While a detailed account of server data migration is beyond the scope of this document, the old server is a valuable data resource that can be mined to drive user provisioning on the Communications Suite servers, as well as changes in mail routing. Sun Professional Services can help you to understand and accommodate server data migration in your deployment plan, and other third-party companies offer both technology and consulting expertise to facilitate the migration of your server data.

Mail Routing During Transition Phase of Server Migration

Not all users will migrate from the old server to the new server at the same time. User accounts on the new server are created and provisioned prior to users' actual migrations. As such, user mailboxes with the same address will reside on both the old and new servers simultaneously during your transition period. Some temporary mail-forwarding rules must therefore be defined to make sure that your users' mail is correctly routed during the transition period.

Even if the organization is implementing new Internet addresses, the old addresses must be maintained on the old server because replies to older messages will be deliverable only if the users' old primary Internet addresses continue to deliver to the correct server. Since all Internet mail for a given domain must go to a single server, specified in an associated MX record, your organization must decide when it will update its MX record to point to the new Sun Java System server.

If the MX record is switched to the Communications Suite server at the beginning of the transition period, you must configure the Communications Suite server so that any mail it cannot deliver to a local mailbox will be sent to the corresponding mailbox on the old server. Moreover, as new users are provisioned on Communications Suite, forwarding rules must be defined on the new server so that any mail that would normally be delivered to the new mailboxes will instead be forwarded to the corresponding user mailboxes on the old server. As each user migrates to the new server, delete the first forwarding rule on the new server, and define a new rule on the old server to forward all of the user's mail to the corresponding Communications Suite mailbox.

On the other hand, if the MX record will point to the old server until the end of the transition phase, configure the old server to send mail it cannot deliver locally to the corresponding user mailboxes on the Communications Suite server. As users migrate to the new server you must then define new rules on the old server to forward mail sent by other users on the old server to the users' new accounts on the Communications Suite server.

Synchronization of Global Address Books During Migration

Larger organizations are likely to require weeks or months to complete a phased migration, and during transition there will be some users on each of the two systems. Many organizations prefer that all users maintain access to an accurate enterprise directory (white pages, global address book), but accuracy will require regular synchronizations of the two servers' directories as employees are hired, transferred, terminated and so forth. The deployment plan should therefore specify some mechanism for regularly synchronizing the two directories throughout your transition period.

Sun Professional Services can assist in addressing this issue. There are also several products available to perform directory synchronization.

New User IDs and Passwords for Sun Java System Servers

Most network systems are designed to prevent people from discovering user passwords, and this is particularly true of Microsoft Exchange. These security safeguards make it impossible to automatically retain users' existing passwords as they migrate from the old server to the new.

Meanwhile, many organizations prefer to standardize the form of their Internet addresses, or to combine domains during their migration to Communications Suite. An organization must therefore decide, in advance, how account names and Internet addresses will be derived, and how users' new passwords will be assigned.

The network administrator must also devise a method for communicating these user credentials to both the users and the corporate help desk. One common approach is simply to prepare an email merge, just prior to a group's migration, so that each member of the migrating group receives his or her credentials independently, and just in time for the group's first logins to the new server.

Where to Go for More Connector for Microsoft Outlook Information

To install administrator's Desktop Deployment Configuration toolkit for the Connector for Microsoft Outlook, see the *Sun Java System Connector for Microsoft Outlook 7.2 Installation Guide*. To configure and administer Connector for Microsoft Outlook, see the *Sun Java System Connector for Microsoft Outlook 7.2 Administration Guide*.

To understand how to configure Calendar Server with Connector for Microsoft Outlook, see *Configuring Calendar Server for Connector for Microsoft Outlook*.

PART VII

Deployment Examples

This part contains the following chapter:

- Chapter 28, "Communications Suite Deployment Examples,"

CHAPTER 28

Communications Suite Deployment Examples

This chapter contains Communications Suite 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 Suite Single-tiered Logical Deployment Example for One Host” on page 313
- “Communications Suite Two-tiered Logical Deployment Example for Multiple Hosts” on page 316

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 81 for more information.

Communications Suite 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 shows a single-host deployment example. The following Communications Suite components are installed on the same host:

- Messaging Server (MTA, Message Store, and Webmail Server)
- 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 Suite. Directory Server and Access Manager are a complex deployment on their own. This figure represents those components by a "cloud."

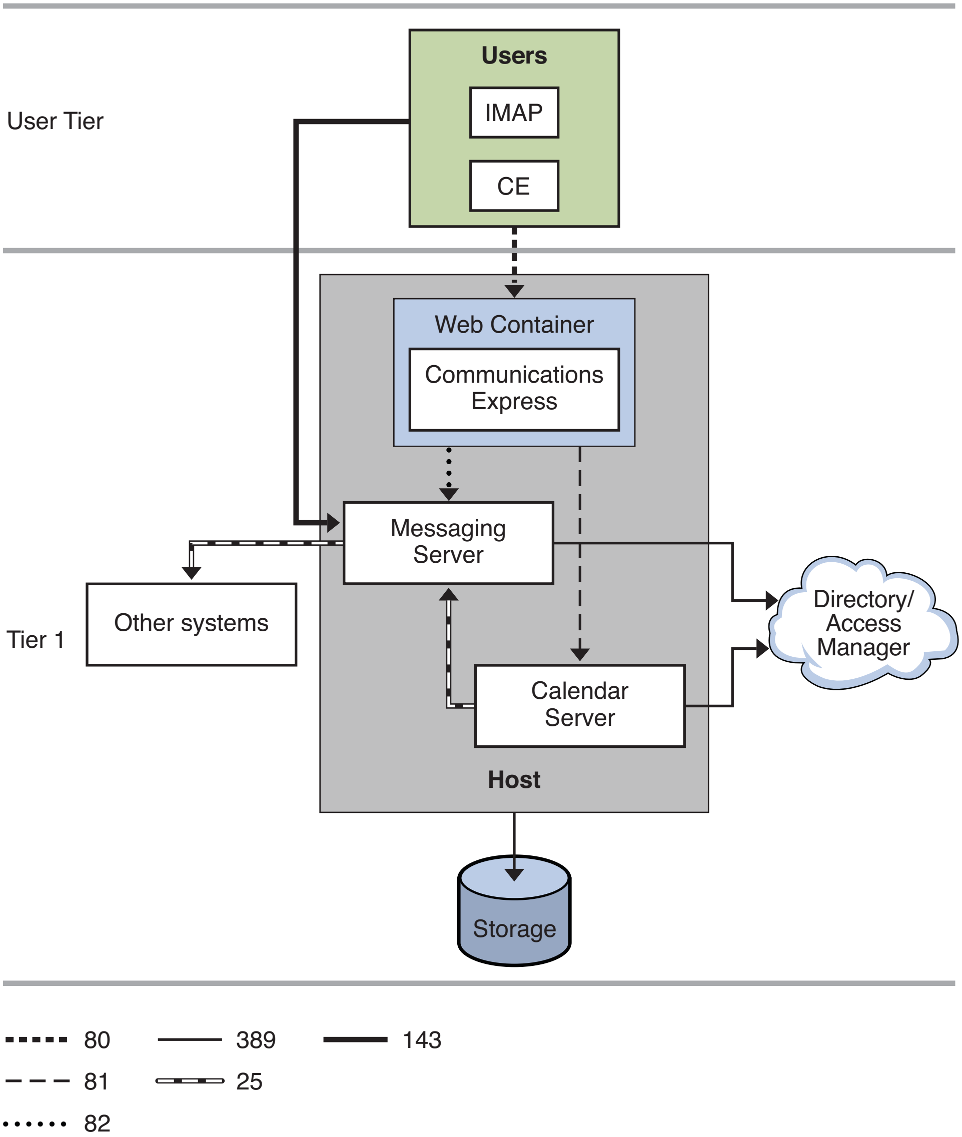

FIGURE 28–1 Communications Suite Single-tiered Deployment Example for One Host

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 (`mshttpd`) components
HTTP	81	Communications Express to communicate with Calendar Server (`cshttpd`)
HTTP	82	Communications Express to communicate with Messaging Server (`mshttpd`)
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 81 for more information. You would consider expanding to a two-tiered architecture when issues arise with capacity, performance, multiple geographic sites, and availability.

Communications Suite Two-tiered Logical Deployment Example for Multiple Hosts

Figure 28–2 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.”

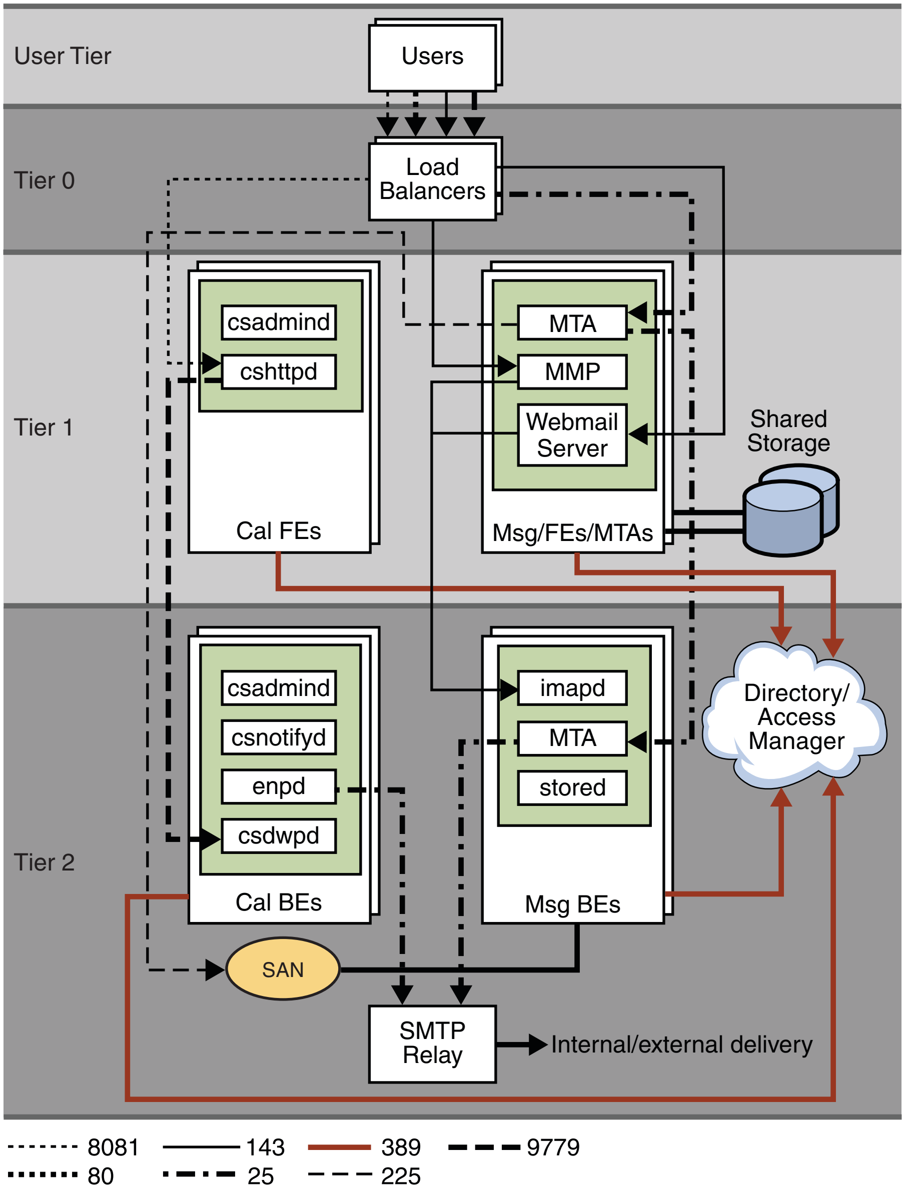

FIGURE 28–2 Communications Suite Two-tiered Deployment Example

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 Webmail (`mshttpd`) 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

Glossary Refer to the *Sun Java Enterprise System Glossary* for a complete list of terms that are used in this documentation set.

Index

A

Access Control Instructions, 164
Access Control Lists, 54
access controls, 184-188, 199, 200, 202
access layer, 78, 258, 259
  components, 49
  corporate directory deployed in, 49
  Directory Server, 73
  Instant Messaging, 257
  network security, 96
  private data network, 149
  single-tiered architecture, 144
  two-tiered architecture, 141
  validating login, 151
Access Manager
  CLI utilities, 104
  deploying, 217, 268-271
  Instant Messaging Service Definition, 238
  integration with Calendar Server, 225
  integration with Instant Messaging, 34
  Portal server and, 271-275
  schema issues, 36, 53, 128
  SSO, 54, 102
  user entry provisioning, 117
ACIs, 164
address book, 287, 288, 291
  Communications Express, 34
  Messenger Express, 57
  portlet, 36
address book lookup, 49
address book module, 291
alerts, 262, 271, 275

alerts *(Continued)*
  Instant Messaging, 237
  overview, 246
anti-spam
  access control, 200
  authentication service, 200
  comprehensive tracing, 200
  deployment issues, 204
  deployment scenarios, 206
  integrating with third-party products, 203
  mailbox filtering, 200
  message address verification, 200
  overview, 199-203
  Real-time Blackhole List, 200
  relay blocking, 200
  sidelining, 200
anti-virus
  conversion channel, 200
  deployment issues, 204
  deployment scenarios, 206
  integrating with third-party products, 203
  overview, 199-203
applet URL, 264, 270
application security, 97-99
architectural strategies, 141, 177, 257-260
architecture
  Instant Messaging one-tiered, 259
  one-tiered, 142, 258, 259
  one-tiered sizing considerations, 141
  two-tiered, 75, 142, 258, 259
  two-tiered and additional MMPs or MTAs, 141
  two-tiered Instant Messaging, 257

architecture *(Continued)*
two-tiered sizing considerations, 141
two-tiered sizing considerations for Instant Messaging, 257-259
ASR, 84, 158
assessing security needs, 94
authentication
Access Manager and, 269-271
DWP and, 226
LDAP and, 239
load on Calendar Server, 58
planning for users, 192-195, 282-283
Portal Server and, 273-275
protecting the Message Store, 191
SASL, 193, 202
server security, 95
SMTP, 194
SSL, 195, 227
users and, 239
validating roaming users, 207
authentication services for Messaging Server, 202
automatic system reconfiguration, 84

B

backup and recovery, 138
balancing network connections, 149
benefits
Calendar Server, 212
Communications Suite summary, 34-35
directory server replication, 84
one-tree DIT structure, 51-53
single-tiered architecture, 77
two-tiered architecture, 77-79
Brightmail, 189, 204, 206
business requirements, 29
defining, 42
understanding, 41

C

calendar alerts, 238, 246, 265
Instant Messaging, 31
calendar event notification, *See* event notification
Calendar Server
Administration Service, 229
and Directory Server interaction, 56
and high availability, 86
and Instant Messaging, 240
authentication, 58
back-end system, 230
benefits, 212
Calendar Express, 59, 214
Calendar Express and Portal Server, 36
configuration examples, 219
configuring Calendar Express to avoid latency delays with Directory Server, 56
considerations, 57-59, 229-231
creating a deployment team, 214
deprecation of Calendar Express, 31
Distributed Database Service, 229
end users, 214-215
Event Notification Service, 229
high availability, 58
HTTP Service, 229
LDAP data cache, 231
Master/slave LDAP configuration, 232
monitoring your system, 226
Notification Service, 229
overview, 31
password, 226
planning for user authentication, 226
plugins, 212
satisfying business needs, 34
security, 226
services, 229
support for industry standards, 211
Calendar Store, 49
capacity planning, 55-56, 77
increases in message traffic, 45
planning for growth, 45-46
central topology, 166, 167, 170, 172
CERT, 100
certificate, 195, 196, 227
and authenticated SMTP, 194
and Calendar Server, 227
implementing two different CAs, 98

certificate *(Continued)*
- login to POP or IMAP, 118
- SSL-based authentication, 194

certificate authorities, 98-99, 195
changes in DIT structure, 50
channel programs
- master, 122
- slave, 122, 124

channels
- configuring, 122
- conversion, 121, 188
- default, 121
- defragmentation, 121
- filter, 188
- keywords, 122
- LMTP, 121
- local, 121
- overview, 121
- pipe, 121
- reprocessing, 121
- SMTP, 121

chat, 237, 246, 253
- Instant Messaging, 31

cipher, 197
CLI utilities
- Access Manager, 104
- Calendar Server, 104
- Message Store, 127
- MTA, 118
- Sun ONE Delegated Administrator, 109

client access filter, 189
`cn` attribute, 239
Communications Express
- address book, 287, 291
- address book module, 291
- overview, 32
- product features, 288
- satisfying business needs, 34
- scaling, 59
- user preferences, 288

Communications Suite
- and Portal Server, 36-37
- architecture, 303
- business requirements, 29

Communications Suite *(Continued)*
- components, 49
- deployment process, 37-40
- examples, 291, 313
- high availability, 35-36
- logical architectures, 69-79
- overview, 29-33
- satisfying business needs, 33-37
- summary of benefits, 34-35

Communications Sync, 32
compatibility mode, Schema 2, 103, 105
component product planning aspects, 47-49
comprehensive tracing, 202
Computer Emergency Response Team, 100
concurrent connections, 132
conference rooms, 212, 237, 245, 246
- Instant Messaging, 31

Connector for Microsoft Outlook
- installation considerations, 305-308
- overview, 32
- product features, 300-301

conversion channel, 121, 188, 203
corporate directory, 49
CPU requirements, 141
CRAM-MD5, 191, 193

D

data layer, 70, 259, 289
- components, 49
- Instant Messaging, 257
- network security, 96
- private data network, 149
- single-tiered architecture, 144
- two-tiered architecture, 141, 147

Database Wire Protocol, 219, 220
- minimal configuration, 218

`db_stat` command, 161
DC tree, 50, 53, 102, 104
defining
- Instant Messaging site profile, 253
- Instant Messaging user base, 253
- Messaging Server user base, 134-135

defragmentation channel, 121

Delegated Administrator, 111, 120
- and Calendar Server, 109, 110
- and Messaging Server, 107
- and Schema 2, 103, 105, 117
- description, 150
- provisioning options, 106
- Schema 1 and, 54, 117

demilitarized zone, 66-67, 74, 174

denial of service, 184, 259
- and multiplexing services, 144
- and single-tiered architecture, 77, 144
- and sizing MTAs, 143
- and two-tiered architecture, 78, 142
- security strategy, 94
- using MTA inbound relay, 149

deployment
- considerations, 80-82
- examples, 261, 291, 313
- goals, 41-45
- process, 37-40

determining project goals, 45-46

Digest-MD5, 191, 193

Directory Information Tree, 50, 54, 127

Directory Proxy Server, 87, 88

Directory Server
- and Personal Address Book considerations, 56-57
- capacity planning, 55-56
- considerations, 54-57
- description, 127
- high availability and, 84-85
- replica role promotion, 88
- replication ability, 128
- tiered architecture considerations, 55
- topology considerations, 55

disaster recovery, 134, 138, 253
- and SLAs, 44

disk
- calculating throughput, 137
- determining amount, 137
- I/O operations per second, 137

disk stripe width, 161

Distributed Database Service, 229

distributed topology, 168

DIT, 50, 54, 127

DIT structure, 50, 51-53

DMZ, 66-67, 74

DNS, 63, 66, 70, 190
- coping with extra load, 44

DNS server, purpose, 120, 149

domain
- hosted, 35, 116
- rewrite rules, 123
- structure, 50
- vanity, 116

DWP, 219, 220, 318
- minimal configuration, 218

E

edge logical architecture, 70, 75-77, 77

EHLO, 99

email
- Instant Messaging and, 240
- notification, 265-267

embedded URLs, 246

encrypted mail, 194, 196, 227

end-user filter, 188

end-user performance, 215

equation to calculate system availability, 45

event notification, 240

Event Notification Service, 229

existing infrastructure, 44

Extensible Messaging and Presence Protocol, 31

F

filtering
- client access, 118
- for end-user, 188
- for unsolicited mail, 118
- `FROM_ACCESS`, 185
- `MAIL_ACCESS` mapping table, 185
- mapping table, 184
- `ORIG_MAIL_ACCESS` mapping table, 185
- `ORIG_SEND_ACCESS`, 185
- `PORT_ACCESS`, 185
- `SEND_ACCESS` mapping table, 185

financial requirements, defining, 44
firewall
  configuration, 68
  DMZ segmentation, 67
  network address translation, 99
  purpose, 64
freeware mail filter, 189
FROM_ACCESS, 185

G

gateway, 177, 225, 241
  legacy messaging systems, 177

H

heavyweight POP users, 135
high availability
  automatic system reconfiguration, 84
  Calendar Server, 58, 86
  Communications Suite, 35-36
  description, 155
  Messaging Server, 86
  overview, 83-84
horizontal scalability, 79-80, 152, 219
hosted domains, 116
HTTP Service for Calendar Server, 229
hybrid topology, 170

I

Identd callback, 190
identifying deployment goals, 41-45
iim.conf file, 252, 255
iim.jvm.maxmemorysize parameter, 255
iim_mux.maxsessions parameter, 252, 257
iim_mux.maxthreads parameter, 257
iim_mux.numinstances parameter, 252, 257
IMAP, 19, 131, 202
  active users, 131
  and access control, 189
  and Message Store performance, 156

IMAP *(Continued)*
  and MMPs, 192
  and SSL, 195
  and topology elements, 180
  certificate-based login to, 118
  clients, 150, 186
  concurrent connections, 143
  internal user scenario, 151
  logical service names, 81
  login using plain text or encryption, 192
  Message Store, 120
imbrand.jar file, 239
imdesktop.jar file, 239
imjini.jar file, 239
imnet.jar file, 239
implementing secure connections, 97-98
imres.jar file, 239
imsbackup command, 138
imta.cnf file, 123
inbound and outbound message traffic, 154
INBOX, 125, 126
incoming messages
  MTAs, 154
  relays, 173
  routing by MTA, 120
index.html file, 239
instant message
  embedded URLs in, 246
  structure, 241-242
Instant Messaging
  Access Manager and, 268-271
  Access Manager requirements, 240
  Access Manager SDK requirements, 240
  alerts, 31, 237, 271, 275
  alerts and basic functionality, 262
  API, 238
  architectural strategies, 257-260
  archive, 255, 256, 271-275
  as a channel in Portal Server, 240
  basic architecture, 242
  basic deployment, 263-265
  Calendar Server requirements, 240
  casual users, 253
  chat, 237, 246

Instant Messaging *(Continued)*
- collecting sizing data, 250-253
- conference rooms, 237, 245, 246
- considerations, 59
- core components, 238, 244-245
- CPU resources, 256-257
- database, 239
- defining usage profile, 250-253
- deploying multiple servers, 277
- deployment examples, 261
- direct LDAP lookup, 245
- disk capacity, 255-256
- disk throughput, 255
- email and, 240
- heavy users, 253
- Instant Messenger, 245-246
- LDAP requirements, 239-240
- load balancers, 259-260
- load simulator, 254
- maximum memory for JVM, 255
- memory utilization, 254-255
- message delivery, 245
- Messaging server and, 265-267
- multiplexor, 245
- network throughput, 256
- news channels, 246
- offline message forwarding, 240
- overview, 31
- overview of alerts, 246
- peak volume defined, 250
- performance guidelines, 254-257
- platform options, 261
- polls, 237, 246
- Portal Server and, 271-275
- Portal Server requirements, 240-241
- Portal Server Secure Remote Access requirements, 241
- protocols, 238
- proxy, 49
- related components, 238-241
- resource requirements, 260
- satisfying business needs, 34
- security, 279
- server defined, 238

Instant Messaging *(Continued)*
- site policies, 283
- SMTP requirements, 240
- SMTP server and, 265-267
- supported standards, 241-242
- tuning `iim.conf` settings for multiplexor, 252
- tuning rules of thumb, 257
- user base defined, 253
- web server requirements, 239
- Webmail, 77

Instant Messaging deployment
- Access Manager and, 268-271
- authentication and, 269-271, 273-275
- email notification and, 265-267
- multiple Instant Messaging servers, 277
- physical, 275-277
- Portal server and, 271-275
- SMTP server and, 265-267

Instant Messaging files
- `iim.conf`, 255
- `imbrand.jar`, 239
- `imdesktop.jar`, 239
- `imjini.jar`, 239
- `imnet.jar`, 239
- `imres.jar`, 239
- `index.html`, 239

Instant Messaging multiplexor
- best practices, 257
- configuration, 257
- core component, 238
- deploying multiple, 276-277

Instant Messenger, 245-246
- applet URL, 264, 270
- communication modes, 246
- non-secure mode, 241
- secure mode, 241

Instant Messenger resources
- files, 239
- installing, 239
- Web server and, 239

internal network, 67

Internet Relay, 173, 180

intranet, 67

IP-spoofing, protecting against, 68

ISDN, 63

J

Java plugin, 264, 271, 274
Java Web Start, 246, 274
    and Instant Messaging, 245
    invoking with Instant Messaging, 264, 271
JavaScript, 79, 289
JavaServer Pages, 289
junk mail, 184, 186

L

LDAP
    attributes affecting master/slave configuration, 232
    attributes for Instant Messaging, 239
    CLD plugin, 215
    considerations, 55
    data cache and Calendar Server, 231
    deployment culture and politics, 42
    DIT requirements, 50-53
    LMTP and, 80
    load balancers and, 88
    Master Server, 56
    one-tree structure, 50
    provisioning, 106-109, 110, 111, 117
    provisioning tools, 109
    user attributes, 239
    user authentication and, 239
    using Calendar Server API for non-LDAP directory server, 218
LDAP database
    Messaging Server, 115
    new groups, 120
    new users, 120
LDAP directory
    for user provisioning, 117
    lookup, 123
    purpose, 120, 149
    tools, 107
LDAP Schema 1, 53, 54, 117
LDAP Schema 2, 53, 117
Legato Networker, 138
lightweight IMAP users, 135
lightweight POP users, 134
limiting network access, 96
limiting physical access to hardware, 95
Linux, default base directory for, 22
LMTP, 78, 80, 318
load balancer, 88, 230
load balancing, 65, 259
    affects on Messaging Server architecture, 116
    algorithms that are load-aware by CPU utilization, 77
    and Calendar Server, 215
    and Messaging Server, 154
    avoiding single points of failure, 88
    for Directory Server, 84
    overview, 65
    simplifying end-user configurations, 78
    using MTAs, 143
load simulator
    Instant Messaging, 254
    Messaging Server, 136
local channel, 121
Local Mail Transfer Protocol
    implementing, 80
    multi-tier deployment, 125
    offloading SMTP processing, 78
log file directory, Message Store, 158
logical architecture
    edge, 75-77
    single-tiered distributed, 72-73
    single-tiered for multiple hosts, 71
    single-tiered for one host, 70-71
    two-tiered, 73-75
logical service names, 81-82

M

Mail Abuse Protection System, 189
MAIL_ACCESS mapping table, 185, 186
mail filter, 187
    client access, 189
    conversion channel, 188
    DNS lookup, 190

mail filter *(Continued)*
  for a channel, 188
  for a user, 187
  for mailbox, 187
  for Message Transfer Agent, 188
  Identd callback, 190
  MAPS RBL, 189
  server-side rules, 187
  Sieve, 188
  SpamAssassin, 189
mailAlternateAddress attribute, 116
mailbox
  database files, 157
  Message Store and, 125
  Messaging Multiplexor and, 150
  optimization, 160
  setting database cache size, 160
mailbox filtering, 200, 201
mailEquivalentAddress attribute, 116
mapping tables to filter mail, 184
MAPS RBL, 80-81
master programs, channels, 121
Master/slave LDAP, 232
mboxlist directory, 158
mediumweight IMAP users, 135
mediumweight Messenger Express users, 135
memory
  large Instant Messaging deployment, 260
  minimum requirement for Instant Messaging, 254
  minimum requirement for Messaging Server, 137
  small Instant Messaging deployment, 260
message address verification, 201
Message Store, 49
  configuring, 118
  definition, 120
  description, 125
  determining size, 142
  folders, 125
  HTTP, 120
  IMAP, 120, 125
  IMAP server, 149
  log file directory, 158
  Messaging Multiplexor, 150
  partitions, 126, 159

Message Store *(Continued)*
  performance issues, 156
  POP, 120, 125
  POP server, 149
  recovering lost data, 126
  scaling, 160
  security issues, 191
  single-copy message store, 126
  sizing, 137
  system files, 126
  user mailboxes, 125
Message Transfer Agent
  access component, 49
  configuring, 118
  incoming email messages, 149
  Internet-facing, 174
  mail filter, 188
  managing messages, 120
  MX records, 154
  outgoing email messages, 149
  performance issues, 162
  protecting relays, 183
  purpose, 120, 121
  queue directories, 158
  service dispatcher, 186
  single-tiered architecture, 71
  SMTP protocol, 120
  two-tiered architecture, 73
Messaging Express Multiplexor, 73
Messaging Multiplexor, 49, 73, 150
  high availability, 163
  performance issues, 163
Messaging Server
  and high availability, 86
  authentication services, 202
  considerations, 57
  determining concurrent connections, 132
  dividing clients across multiple hosts, 152
  firewall, 67
  hosted domains, 116
  LDAP database, 115, 117
  load balancing, 65, 154
  Local Mail Transfer Protocol, 125
  log file directory, 158

Messaging Server *(Continued)*
- message load for a server machine, 155
- Message Transfer Agent, 71, 73
- Messaging Express Multiplexor, 73
- Messaging Multiplexor, 71
- mobile users, 68
- overview, 30
- performance issues, 156
- satisfying business needs, 33
- Schema 1, 128
- Schema 2, 128
- Secure Sockets Layer, 194, 196
- security features, 118
- sizing Message Store, 137
- spam, 78
- standalone view, 119
- supported protocols, 116
- two-tiered architecture, 141, 142, 147
- unified messaging solution, 118
- Webmail, 98

messaging topology, 165
Messenger Express, 149, 188, 291
- and Personal Address Book, 56
- as part of tiered architecture, 70, 71, 72, 74
- authentication protocols, 192
- client connections, 131
- encryption, 197
- mail filters, 108, 187
- mailbox filtering, 201
- making secure, 96
- S/MIME, 197
- security considerations, 192
- security issues, 191
- signed messages, 197
- storing mail filters, 187

`messenger.jar` file, 239
Microsoft Outlook, 32, 72, 74, 214
- connector, 49

milter, 203
MIME messages, 121
MMP, 150
- security issues, 191

MMP SMTP Proxy, 176
mobile users, 68
monitoring your system, 190, 226
MTA router, 143
MX records, 154

N

Netlet, secure mode, 241
network
- balancing connections, 149
- demilitarized zone, 66
- firewall, 64
- infrastructure components, 64-66
- internal, 67-68
- limiting access, 96
- mobile users, 68
- planning infrastructure layout, 66-68
- proxy, 68
- routers, 64
- switch, 64
- throughput issues, 140
- Virtual Private Networking, 170
- WAN, 89, 170

network address translation, 99
news channels, 246
non-secure mode,Instant Messenger, 241
NOTARY, 99
Notification Service, 229

O

offline message forwarding, 240
one-tiered architecture
- description, 144
- Instant Messaging, 259
- security issues, 191

operating system security, 95-96
operation requirements, 42
Organization Tree, 50, 102, 104
`ORIG_MAIL_ACCESS` mapping table, 185
`ORIG_SEND_ACCESS`, 185
outbound and inbound message traffic, 154
outgoing mail messages, 120

overview
- Calendar Server, 31
- Communications Express, 32
- Communications Suite, 29-33
- Communications Sync, 32
- Connector for Microsoft Outlook, 32
- Instant Messaging, 31
- logical architectures, 69-79
- Messaging Server, 30

P

parameters, `iim.jvm.maxmemorysize`, 255
password
- CRAM-MD5, 193
- encrypted, 192, 226
- issues, 192, 226, 282
- plain text, 192, 226
- SASL, 193

peak volume
- defining for Instant Messaging, 250
- determining for Messaging Server, 130

performance
- guidelines for Instant Messaging, 254-257
- Message Store, 156
- Message Transfer Agent, 162
- Messaging Multiplexor, 163
- Messaging Server issues, 156

Personal Address Book, 56-57, 149
physical access to hardware, 95
physical deployment examples, 275-277
pipe channel, 121
policy management, 268-271
polls, 237, 246
- Instant Messaging, 31

POP, 156, 163, 180
- access policy, 133
- active users, 131
- and Message Store, 120
- certificate-based login, 118
- connections during peak volume, 131
- internal user scenario, 151
- plain text and encrypted login, 192
- SASL, 202

POP *(Continued)*
- security, 186, 189, 192
- sizing multiplexing services, 143
- SSL, 195
- user facing logical service names, 81, 150

`PORT_ACCESS`, 185, 186
port numbers
- Communications Express remote host deployment example, 292, 294
- single-tiered Communications Suite deployment example, 316
- two-tiered Communications Suite deployment example, 318

Portal Server
- Access Manager and, 271-275
- and Communications Suite, 36-37
- considerations, 59
- Desktop, 36, 240
- Instant Messaging deployment and, 271-275
- Netlet, 241

premium broadband users, 135
private backup network, 255
private key, 194, 227
program channels, master, 124
project goals, 45-46
protecting MTAs, 183
protecting software, 95
protocols
- public, 148
- supported by Messaging Server, 116

provisioning options
- deciding on schema version, 101, 104
- Delegated Administrator, 106
- LDAP directory tools, 107, 110
- provisioning tools, 106, 109
- tool comparisons, 107, 110

provisioning users, 117
proxy, 68
public access protocols, 148
public key, 194, 227

R

Real-time Blackhole List, 80-81, 189, 201

reducing hardware downtime, 84
relay blocking, 202
remote site failover, 89, 91
replica role promotion, 88
reprocessing channel, 121
requirements
  business, 42
  culture and politics, 42
  financial, 44
  Instant Messenger resources, 239
  operation, 42
  schema, 53-54
  technical, 42-44
residential dial-up, 134
retrieving mail, internal user, 151
reverse DNS, 78, 189
rewrite rules
  example, 123
  message header, 123
  purpose, 123
  transport layer, 123
roaming users, 207
routers, 64

S

S/MIME, 97
S/MIME messages, 197
SAML, 97
SAN, 65
SASL, 193
scaling
  Communications Express, 59
  front-end and back-end services, 79-80
  Messaging Store, 160
schema
  choosing, 101
  requirements, 53-54
Schema 1, 36, 54, 117
  Delegated Administrator, 102, 104, 106
  description, 102, 104
  Messaging Server support, 128
  requirements, 53
Schema 2, 36, 53
Schema 2 *(Continued)*
  Access Manager, 107, 110
  choosing, 102, 104
  compatibility mode, 103, 105-106
  Messaging Server support, 128
  native mode, 102, 105
  requirements, 53
  support, 117
schema versions, choosing, 104
secure connections, 97-98
secure mode, Instant Messenger, 241
Secure Sockets Layer, 227
  access control, 118
  cipher, 197
  encryption, 196
  hardware accelerator, 163
  Messaging Server, 194
  SASL, 193
security
  application, 97-99
  assessing needs, 94
  cipher, 197
  Computer Emergency Response Team, 100
  CRAM-MD5, 193
  Digest-MD5, 193
  encryption, 196, 197
  hiding IP addresses, 99
  hiding product names and versions, 99
  Instant Messaging components, 279
  limiting access to network, 96
  limiting physical access to hardware, 95
  message store, 191
  Messenger Express, 191
  misconceptions, 99
  MMP, 191
  network address translation, 99
  not using EHLO, 99
  NOTARY, 99
  one-tiered architecture, 191
  operating system, 95-96
  password, 192, 282
  protecting Instant Messaging server and multiplexor, 280
  protecting MTA relays, 183

security *(Continued)*
- protecting software, 95
- protocols, 118
- SASL, 193
- Secure Sockets Layer, 193, 194, 196, 227
- server, 95
- signatures, 197
- SMTP AUTH, 191, 194
- TLS negotiation, 99
- two-tiered architecture, 191

security features, 118

`SEND_ACCESS` mapping table, 185, 186

sending mail
- internal user to internal user, 150
- internal user to internet user, 151
- internet user to internal user, 151

server-side rules, 187

service level agreements, defining, 44-45

service names, 81-82

service provider topology, 172

service tiers, understanding, 48-49

sidelining, 202

Sieve, mail filter, 188

Simple Authentication and Security Layer, 118

Simple Mail Transport Protocol
- authenticated, 194
- channel, 121
- gateway, 177
- purpose, 120
- SMTP AUTH, 191

single-copy message store, 126

single sign-on, 54, 268-271

single-tiered architecture
- benefits, 77
- distributed, 72-73
- for multiple hosts, 71
- overview, 69

site distribution, 43

site failover, 253

site profile, 253

sizing
- collecting Instant Messaging data, 250-253
- collecting Messaging Server data, 130

sizing *(Continued)*
- components in single-tiered or two-tiered architectures, 75
- Instant Messaging multiplexors, 258-259
- Message Store, 137
- MMP, 163

SLAs, defining, 44-45

slave program, channels, 122

SMTP AUTH, 186, 194

snapshot function, 126

Solaris Fingerprint Database, 96

Solaris Security Toolkit, 65, 96

Solid State Disks, 158

Solstice Backup, 138

Solution Life Cycle, 37-40

spam
- Brightmail products, 189
- Messaging Server, 78
- protecting against, 188
- SpamAssassin, 189

SpamAssassin, 204, 206-207

spoofing, 184

SSL, 78, 97, 98
- accessing data stores, 73
- edge architecture, 76

SSL/TLS, 97

SSO, 54

standards, supported by Instant Messaging, 241-242

Storage Area Network, 49, 65

`store.dbcachesize` parameter, 160, 161

strategy for scalability, 79-80

structure, instant message, 241-242

Sun Cluster software, 58, 230
- and Directory Server, 55, 85
- and Instant Messaging, 86
- and Messaging Server, 83
- and remote site failover, 91
- high availability, 36

Sun ONE Synchronization, 214

support personnel, 44

supporting existing usage patterns, 42-43

switches, 64
- coping with extra load, 44

Symantec AntiVirus Scan Engine, 207

Symantec Brightmail, 204, 206
sync tool, 32

T

TCO, 46
technical requirements, 42-44
TLS negotiation, 99
tools
  comparisons, 107, 110
  provisioning, 106, 109
topology
  central, 166
  components, 173
  designing, 165
  distributed, 168
  example, 177
  hybrid, 170
  service provider, 172
total cost of ownership, 46
Transport Layer Security, 118
Tripwire for Servers, 96
two-tiered architecture, 73-75, 77-79, 257-259
  access layer, 147
  advantages, 142
  data layer, 147
  description, 141
  overview, 69
  performance, 191
  questions about, 43
  separate servers, 70

U

`uid` attribute, 239
unauthorized mail relaying, 184
understanding
  business requirements, 41
  Connector for Microsoft Outlook pre-installation requirements, 305
  Messaging Server topology elements, 173
  remote site failover, 89
  schema and provisioning options, 101
understanding *(Continued)*
  service tiers, 48-49
uptime calculation, 45
URL
  embedded in instant message, 246
  Instant Messaging applet, 264
usage patterns, 42-43, 54, 143
usage profile
  defining for Instant Messaging, 250-253
  determining for Messaging Server, 130
users
  authenticating in LDAP, 239
  heavyweight POP, 135
  lightweight IMAP users, 135
  lightweight POP, 134
  mediumweight IMAP, 135
  mediumweight Messenger Express, 135

V

vanity domains, 116
`/var/mail` system, 120, 121
Veritas Cluster Server software, 35, 83, 86
Veritas software, high availability, 91
vertical scalability, 155
Virtual Private Networking, 170
virus
  Brightmail products, 189
  protecting against, 188
VPN, 63, 67, 170, 191

W

WAN, 89, 170
web container, 291, 293
  and Communications Express, 287
  for Communications Express, 85
Web Server
  installing with Instant Messaging, 276
  Instant Messenger resources and, 239
Webmail, 158, 316, 318
  and Messaging Server, 71, 98, 118
  sticky bit load balancing, 77

X

XMPP, 31, 34